徐宇杰 编著

# TCP/IP协议深入分析



清华大学出版社



# TCP/IP 协议深入分析

徐宇杰　编著

清 华 大 学 出 版 社
北　京

## 内 容 简 介

本书详细说明 TCP/IP 协议簇，以截屏和协议包结构为手段，介绍了 TCP/IP 各层的细节，对 IP、TCP、UDP、ARP、ICMP、HTTP、Telnet、FTP 与 TFTP、POP3 与 SMTP、DHCP 协议进行了深入剖析。本书实例丰富，图文并茂，注重理论与实践结合，降低了读者的学习难度，激发了读者学习兴趣和动手欲望。

本书可作为网络从业人员的专业学习和参考用书，也可作为大中专院校网络课程教材。



**图书在版编目(CIP)数据**

TCP/IP 协议深入分析/徐宇杰编著. —北京：清华大学出版社，2009. 2
ISBN 978-7-302-18416-4

Ⅰ. T…　Ⅱ. 徐…　Ⅲ. 计算机网络—通信协议　Ⅳ. TN915. 04

中国版本图书馆 CIP 数据核字(2008)第 125636 号

**责任编辑：**丁　岭　赵晓宁
**责任校对：**焦丽丽
**责任印制：**

**出版发行：**清华大学出版社　　**地　址：**北京清华大学学研大厦 A 座
http://www. tup. com. cn　　**邮　编：**100084
**社 总 机：**010-62770175　　**邮　购：**010-62786544
**投稿与读者服务：**010-62776969，c-service@tup. tsinghua. edu. cn
**质 量 反 馈：**010-62772015，zhiliang@tup. tsinghua. edu. cn
**印 刷 者：**
**装 订 者：**
**经　销：**全国新华书店
**开　本：**185×260　**印　张：**10. 25　**字　数：**250 千字
**版　次：**2009 年 2 月第 1 版　**印　次：**2009 年 2 月第1 次印刷
**印　数：**1～　000
**定　价：**　. 00 元

本书如存在文字不清、漏印、缺页、倒页、脱页等印装质量问题，请与清华大学出版社出版部联系调换。
联系电话：010-62770177 转 3103　　产品编号：

FOREWORD

# 前言

Internet不仅深刻地改变了整个IT行业的格局和计算模式，也深刻改变了经商的方式和人们的生活方式，网络已经成为整个基础设施中的重要部分。

Internet也叫网际网，也就是网络和网络互联。因此，当前的网络技术不仅仅是Socket编程这样点对点的通信技术和Windows Server下的Web服务器配置这样的系统管理技术。从TCP/IP的视角来看，所谓网络技术实际上包括计算机网络原理、计算机网络设计、计算机网络工程、计算机网络协议、计算机网络互联、计算机网络应用等几个方面的范畴。一个网络的互联是跨越不同网络层次的一个过程。因此，基于互联网的网络技术的设计、实现、管理和排错，需要自底向上的多层次知识。

作者从20世纪90年代初期就开始学习和从事TCP/IP网络技术，工作涉及局域网、园区网、城域网、跨区域网的设计、实现和管理的全过程，经历了从局域网到互联网那激动人心的变化。又有多年大型网络服务器的管理经验，积累了丰富的网络技术实践经验。作者多年的学习、管理、开发和应用，深感目前计算机网络技术方面的书籍往往只介绍一个方面的技术，顾此失彼的多，导致多层结构的网络技术被支离破碎地介绍，很多技术方面的介绍流于表面，学习者很难透彻了解。理论不深刻，实践不实际的情况非常普遍。同时，很多网络厂商又从各自的市场利益出发，积极推广相关的认证和课程。这些课程对网络技术的应用起到了非常正面的作用。但是，这些课程指导书由于更偏向认证的各个知识点，实验内容也围绕认证服务，对于网络技术的细节缺乏深入分析。

如何兼顾理论和实际应用，一直是困扰计算机网络教学的一个难题。作者多年的实践经验一再证明没有扎实的网络理论知识，根本无法从事网络技术。另一方面，学习者只有通过真实项目的全过程实践才能真正掌握网络理论知识。

笔者5年前开始总结自己学习和实践经验，试图将实际项目中最常用、也是最核心的知识点加以梳理。此系列书籍的选题、内容选择、实验的准备和测试，可以说是去粗取精的结果。本系列丛书涵盖了TCP/IP协议的分析、网络交换、网络

路由、下一代互联网技术。有详细的包结构截屏和深入的细节分析，从基本理论学习和分析开始，逐步分析网络技术的各层细节，最后提供一个从布线开始，包含设计和配置、运行一个真实园区网络的综合实例教程。本系列丛书实例丰富、图文并茂，边讲解边操作，将网络技术的细节一一展现，降低了读者的学习难度，激发了学习兴趣和动手欲望，适合网络从业人员的专业学习和参考以及各个院校作为计算机网络的实例教学。

5年的时间，对于网络技术来说似乎是一个漫长的过程，实际上是一个让时间来检验的优选过程。每本书经过多次修改，不仅进行理论内容的修正，更多的是实验的更新，甚至全部改写，试图在出版前反映最新的技术变化。但有可能百密一疏，望读者指正。

编　者
2008年6月

CONTENTS

# 目录

CHAPTER 1

# 第 1 章 TCP/IP 协议概述

TCP/IP 是一个四层协议系统，TCP/IP 协议族是一组不同的协议组合在一起构成的协议族。如表 1-1 所示，每一层负责不同的功能。

**表 1-1**

| TCP/IP | 主要协议 | 主要功能 |
|---|---|---|
| 应用层 | Http、Telnet、FTP、E-mail 等 | 负责把数据传输到传输层或者接收从传输层返回的数据 |
| 传输层 | TCP、UDP | 为两台主机上的应用程序提供端到端的通信。TCP 为两台主机提供高可靠性的数据通信，它所做的工作包括把应用程序交给它的数据分成大小合适的数据块交给下面的网络层，确认接收到的分组等。UDP 则为应用层提供不可靠的数据通信，它只是把数据包的分组从一台主机发送到另一台主机，但是不保证该数据能到达另一端 |
| 网络层 | ICMP、IP、IGMP | 主要为数据包选择路由，其中 IP 是 TCP/IP 协议族中最为核心的协议，所有的 TCP、UDP、ICMP、IGMP 数据都以 IP 数据包格式传输 |
| 链路层 | ARP、RARP 和设备驱动程序及接口 | 发送时将 IP 包作为帧发送，接收时把收到的位组装成帧，同时提供链路管理，错误检测等 |

TCP/IP 协议族中 TCP 和 IP 只是其中的两种协议，其中 TCP 和 UDP 是两种最为著名的传输层协议，IP 是网络层协议。IP 和 TCP 这两个协议的功能不尽相同，它们是在同一时期作为一个协议来设计的，并且在功能上也是互补的，虽然它们可以分开单独使用，但是只有两者的结合，才能保证 Internet 在复杂的环境下正常运行。要连接到 Internet 的计算机，都必须同时安装和使用这两个协议，因此在实际中常把这两个协议统称作 TCP/IP 协议。

在 TCP/IP 协议族中，各层的关系如图 1-1 所示。

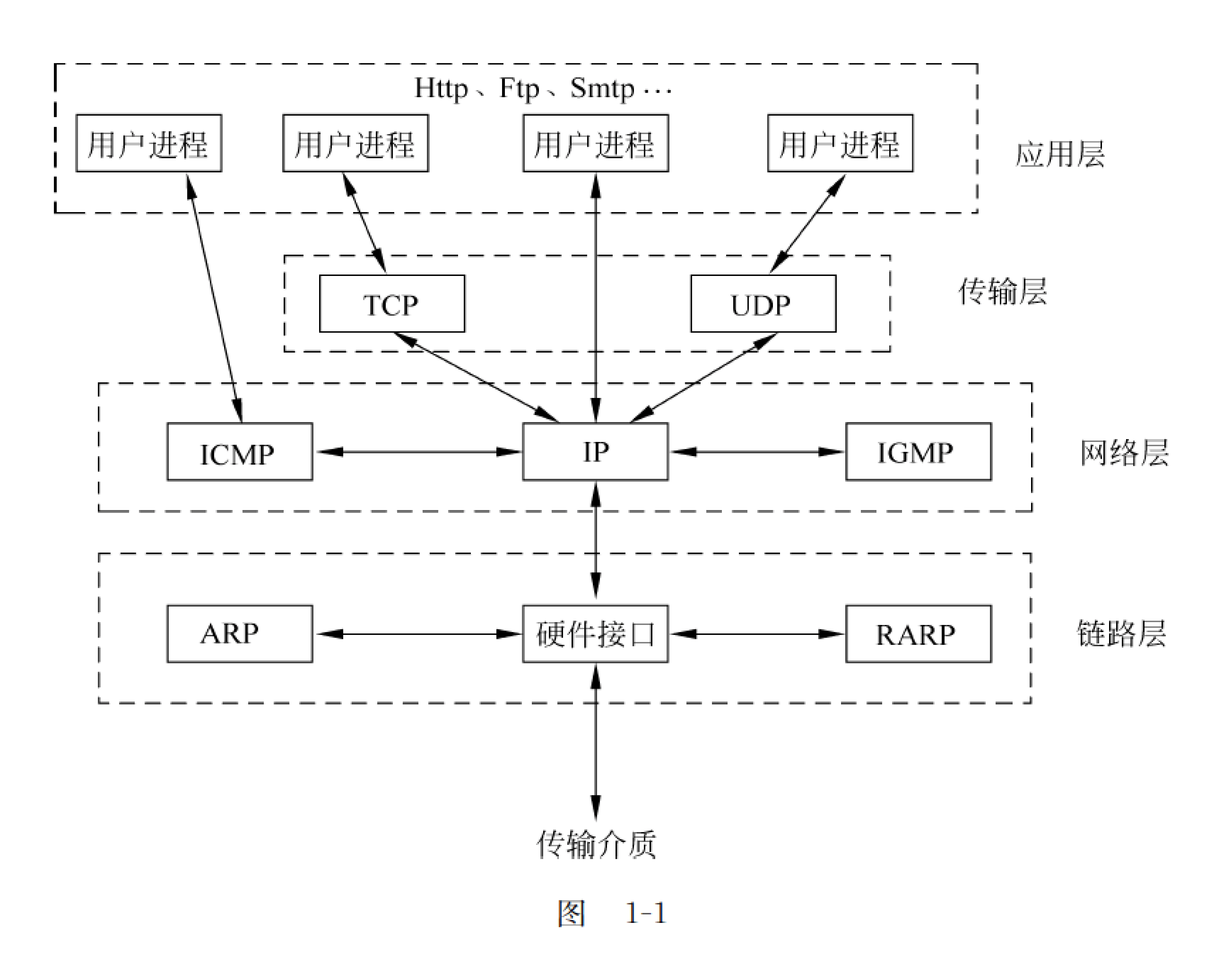



图 1-1

因特网控制消息协议(Internet Control Message Protocol,ICMP)是 IP 协议的附属协议。IP 层用它来与其他主机或路由器交换错误报文和其他重要信息。IGMP 是 Internet 组管理协议,它用来把一个 UDP 数据包多播到多个主机。

ARP(地址解析协议)和 RARP(逆地址解析协议)是某些网络接口(如以太网和令牌环网)使用的特殊协议,它们用来转换网络接口的物理地址和对应的 IP 地址。

当目的主机收到一个以太网数据帧时,数据就开始从协议栈的底部往上升,同时去掉各层协议封装的报文首部。每层协议盒都要去检查报文首部中的协议标识,以确定接收数据的上层协议。这个过程称作分用(Demult IP Lexing),如图 1-2 所示。

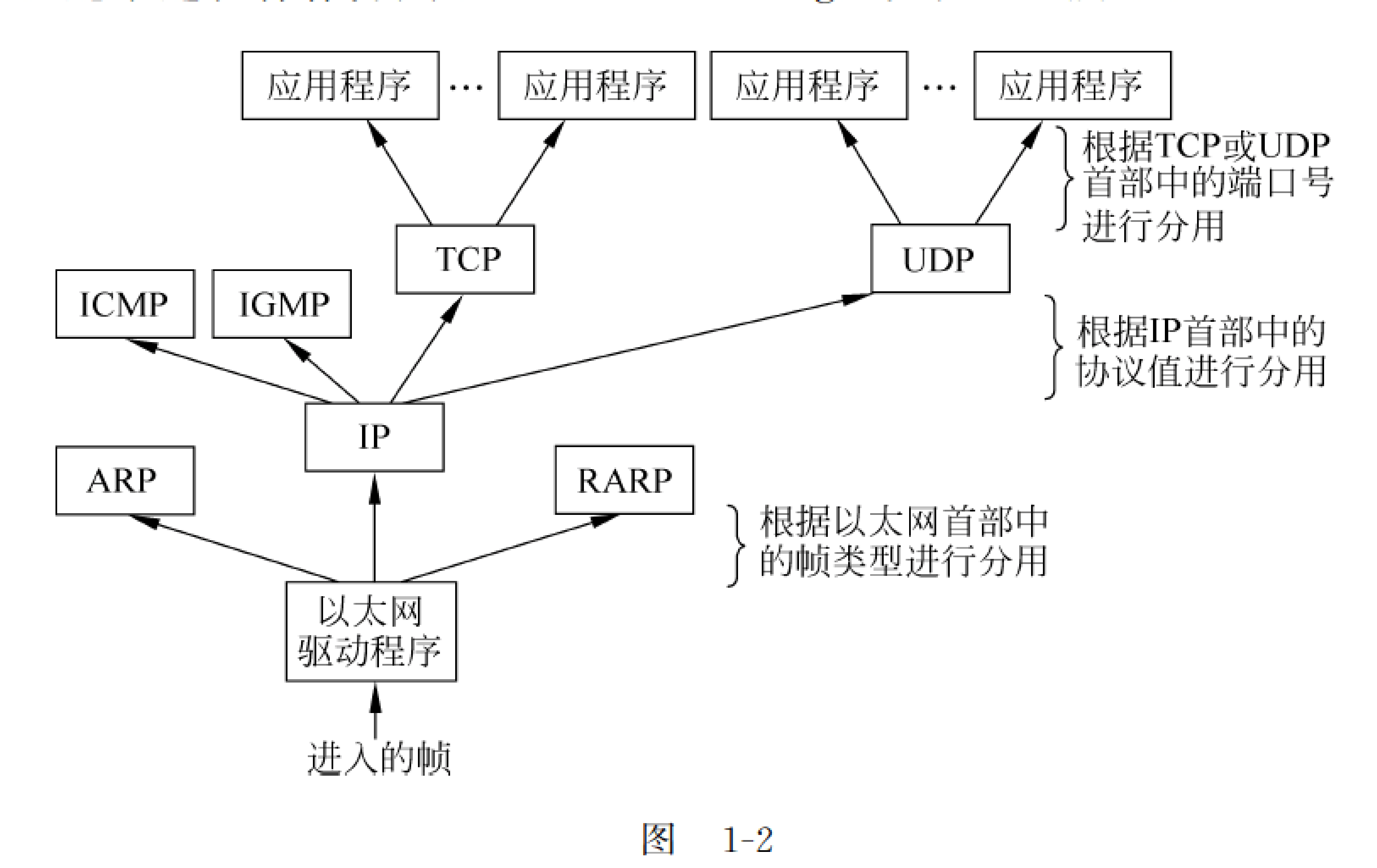



图 1-2

图 1-1 和图 1-2 中的协议分层并不是绝对的,拿 ICMP 和 IGMP 来说,在图 1-1 中,把它们与 IP 放在同一层上,那是因为事实上它们是 IP 的附属协议。但是在图 1-2 中,又把它们

放在 IP 层的上面，这是因为 ICMP 和 IGMP 报文都被封装在 IP 数据包中。

再比如 ARP 和 RARP，在图 1-1 中，把 ARP 作为以太网设备驱动程序的一部分，放在 IP 层的下面。在图 1-2 中，把它们放在以太网设备驱动程序的上方，这是因为它们和 IP 数据包一样，都有各自的以太网数据帧类型。

这里用两种图只想说明各种协议之间彼此的关系，它们之间并不是彼此孤立的。

### 1. IP 层

在 TCP/IP 协议族中，网络层 IP 提供的是一种不可靠的服务，它只是尽可能快地把数据从源结点送到目的结点，并不提供任何可靠性保证。在通信中，IP 层只负责数据的路由与传输，并不处理数据包的内容。例如 ICMP，TCP 或 UDP，这些协议是依赖 IP 层的传输功能来传送数据的。在通信双方的主机中，收到这些协议的数据包后，一般在通信的对应主机上，会有程序来处理这些数据。

### 2. TCP 层

TCP 层位于 IP 的上层，应用程序在 IP 网络上相互之间传输的标准传输协议有两个，一个是传输控制协议（TCP），TCP 是目前 Internet 上使用的最重要的协议，它提供的是可靠的、可控制的传输服务，大部分 Internet 应用程序都使用 TCP，因为它的嵌入可靠性和流控制服务可确保数据不会丢失和被破坏。另一个是用户数据包协议（UDP），它提供的服务轻便但不可靠。

IP 层提供了一种不可靠的服务，TCP 在不可靠的 IP 层上提供了一个可靠的传输层，TCP 采用了超时重传、发送和接收端到端的数据确认等机制来保证这种服务的可靠性。由此可见，传输层和网络层分别负责不同的功能。

CHAPTER 2

# 第2章 IP 协议

IP 是 TCP/IP 协议族中最为核心的协议。所有的 TCP、UDP、ICMP 及 IGMP 数据都以 IP 数据包格式传输。目前的协议版本号是 4，因此 IP 有时也称作 IPv4。未来的版本就是 IPv6。

IP 层只负责数据的路由与传输，并不处理数据包的内容。IP 提供不可靠、无连接的数据包传送服务。

不可靠(unreliable)的意思是它不能保证 IP 数据包能成功地到达目的地。IP 仅提供最好的传输服务。如果在传输过程中发生某些错误时，IP 有一个简单的处理错误算法：丢弃该数据包，然后发送 ICMP 消息报给源发送端。

每经过一个系统(主机或路由器)都会检查这个包，如果这个包已经损坏或者经历过其他形式的暂时性失败，该包就会在此立刻被删除。

如果出现的问题是半永久性(semi-permanent)的，那么 IP 就会发送 ICMP 来给源发送者返回一个错误消息，将这次失败告诉源发送者，以让发送者修改引起失败的情况，然后删除该数据包。

暂时性的失败和半永久性失败之间的界限很重要。暂时性错误是指不是因为发送者的错误引起的(例如生命期超时发生的错误或者校验和计算错误)。半永久性失败是指包或网络出现了问题导致这条路径不能发送数据，这时最好把问题告诉发送者以便纠正错误。

IP 提供无连接服务也就是 IP 并不维护任何关于后续数据包的状态信息。IP 网络把每个 IP 数据包都当作一个完全独立的实体，每一个实体都通过当时最合适的路径进行传输。例如，如果一个用户想从一个远程 Web 服务器获取一个文档，为了返回请求的材料，这个服务器可能需要创建几个 IP 数据包。每个数据包都通过沿途的路由器来寻找最合适的路径进行传输。比如 Web 服务器往请求客户端发送的第一个数据包可能通过地下光纤电缆传输，而第二个可能通过卫星连接来传输，第三个可能通过传统的网络来传输。每个数据包之间是独立的，它的好处就是数据包不用按顺序到达目的地，因为某个数据包可能经过一个快速网络传输，而其他数据

包可能经过一个慢速网络传输，还有数据包可能会重复，导致某个包有多个拷贝到达目的地。

另外，网络把每个数据包都看作是单个的实体，它自身不用负责跟踪所有的连接，这样网络设备可以专注于传输数据包，这个特性使得整个性能能够提高到硬件允许的水平，而对内存和 CPU 要求却尽可能地低。

由于每个数据包的处理是相互独立的，这样当数据包到达目的地时就会失序，怎么样来组织起这些失序的数据包呢？原来在 IP 首部中包含着一些信息，以让接收端能正确组装这些数据包。

## 2.1 IP 分片和重组

IP 分片与网络的 MTU 有关。先来看看 MTU：

MTU(Maximum Transmission Unit)是网络上传送的最大数据包。MTU 的单位是字节。例如，Ethernet 在一个帧中仅仅能够发送 1500 个字节，而 16MB/s Token Ring 的典型的 MTU 每个帧是 17 914 个字节。

RFC791 规定，MTU 最大是 65 535 个字节，最小是 68 个字节。当 IP 层接收到一份要发送的 IP 数据包时，如果数据包大于发送系统的本地 MTU 的话，这时系统就得把数据包分成多个片(fragmentation)来发送。在数据包传输过程中，如果 IP 数据包太大而不能通过发送者和最终接收者之间的某个网段时，路由器也会把包分段，使得数据包能顺利通过这个网络。

分片后的 IP 数据包，只有到达目的地才进行重新组装。重新组装由目的端的 IP 层来完成，其目的是使分片和重新组装过程对传输层(TCP 和 UDP)是透明的。已经分片过的数据包有可能会再次进行分片(可能不止一次)。

当 IP 数据包被分片后，每一片都成为一个分组，具有自己的 IP 首部，这些分片后的数据包相互独立，会选择各自的最佳路由到达目的地。这样，当数据包的这些片到达目的端时有可能会失序，接收端凭件在 IP 首部中的信息正确组装这些数据包片。

虽然数据包可以分片，但是要尽量避免分片，因为 IP 层本身没有超时重传的机制，由更高层来负责超时和重传(TCP 有超时和重传机制)。当某片报文丢失后，TCP 在超时后会重发整个 TCP 报文段，而不是重传数据包文段中的一个数据包片。分片还会增加丢包率，降低网络速度。

## 2.2 分片规则

分片仅仅出现在数据包的数据部分。

分片过程不包括数据包的首部。如果源数据包是 4464 字节，那么这个数据包中至少有 20 字节是用来储存首部信息，这意味着数据部分是 4444 字节。要分片的就是这 4444 字节。

每一个分片都会产生一个包含它自己 IP 首部的新包，这个 IP 首部至少耗用新包中的 20 字节。

IP 分片必须以 8 字节的倍数分片。如一个数据包包含 256 字节的数据,但前一个片段仅能容纳 250 字节,那么第一个片段仅仅包含 248 字节(248 是小于 250 的可以被 8 整除的整数)。剩下的 8 字节就在下一个片段发送。

要尽量避免 IP 分片,TCP 的 Path MTU Discovery 机制可以提供一个最优值来避免数据分片。

## 2.3 IP 数据包结构

IP 数据包包含要发送的数据和相关的 IP 首部,如图 2-1 所示,这是个 IP 数据包的结构,显示了一个 IP 数据包所包含的信息。

<table>
<tr><td>4位版本</td><td>4位首部长度</td><td>8位服务类型(TOS)</td><td colspan="2">16位总长度(字节数)</td><td rowspan="5">20个字节</td></tr>
<tr><td colspan="3">16位标识</td><td>3位标志</td><td>13位片偏移</td></tr>
<tr><td colspan="2">8位生存时间(TTL)</td><td>8位协议</td><td colspan="2">16位首部检验和</td></tr>
<tr><td colspan="5">32位源IP地址</td></tr>
<tr><td colspan="5">32位目的IP地址</td></tr>
<tr><td colspan="5">选项</td></tr>
<tr><td colspan="5">数据</td></tr>
</table>

图 2-1

IP 数据包中每个字段的意义如表 2-1 所示。

**表 2-1**

| 字　段 | 位数 | 用　法　说　明 |
|---|---|---|
| 版本(Version) | 4 | 说明用以创建该数据包的 IP 版本。所有接触该数据包的设备都必须支持本字段显示的版本。大部分 TCP/IP 产品都使用 IPv4 |
| 首部长度(Header Length) | 4 | 以 32 位为单位表明 IP 首部的长度。因为几乎所有的 IP 首部都是 20 字节长,这个字段的值几乎总是 5 |
| 服务类型(Type-of-Service Flags) | 8 | 给应用程序、主机和 Internet 上的路由器提供一个优先级服务。在这个字段设置合适的标志,应用程序可以要求这个数据包得到高优先级,而让其他数据包等待 |
| 总长度(Total Packet Length) | 16 | 以字节为单位说明全部 IP 包的长度,包括首部和主体部分 |
| 标识(Fragment Identifier) | 16 | 标识数据包,在出现分片并想把片段合并成原状时是有用的 |
| 标志(Fragmentation Flags) | 3 | 说明可能出现的任何分片的某些方面,也提供了分片控制服务,例如不让路由器分片某个包 |
| 偏移(Fragmentation Offset) | 13 | 说明这个片段提供的源 IP 数据包的字节范围,用 8 字节的偏移表示 |

续表

| 字　　段 | 位数 | 用　法　说　明 |
|---|---|---|
| 生存时间(Time-to-Live) | 8 | 说明数据包在不可发送和破坏之前还可以经过的跳数 |
| 协议(Protocol Identifier) | 8 | 说明储存在 IP 数据包主体的高层协议 |
| 首部校验和(Header Checksum) | 16 | 用来储存 IP 首部的校验和 |
| 源 IP 地址(Source IP Address) | 32 | 用来储存最初发送该数据包的主机的 32 位的 IP 地址 |
| 目的 IP 地址(Destination IP Address) | 32 | 用来储存该数据包到达目的系统的 32 位的 IP 地址 |
| 选项(options) | 可变 | 就像 IP 用 type-of service 标志提供了一些优先级服务一样，附加的特殊处理选项能够通过 Options 字段定义。这些选项包括 source routing，timestamp 和其他一些选项。这些选项很少用，这也是会导致 IP 首部长度超过 20 字节的唯一原因(这个选项是可选择的) |
| Padding(如果需要的话) | 可变 | IP 数据包的首部的长度必须是 32 的倍数。如果首部中引入了某些选项，首部必须填充到能够被 32 整除的位数 |
| 数据(data) | 可变 | IP 数据包的数据部分。正常情况下，会包含一个完全的 TCP 或 UDP 信息，但是它也可以是其他 IP 数据包的一个片段 |

下面通过一个实际的 FTP 数据包来分析一下 IP 数据包的结构。

### 1. Version 和 Header Length

如图 2-2 和图 2-3 所示，可以看到 IP 数据包前面两个字段 Version 和 Header Length。

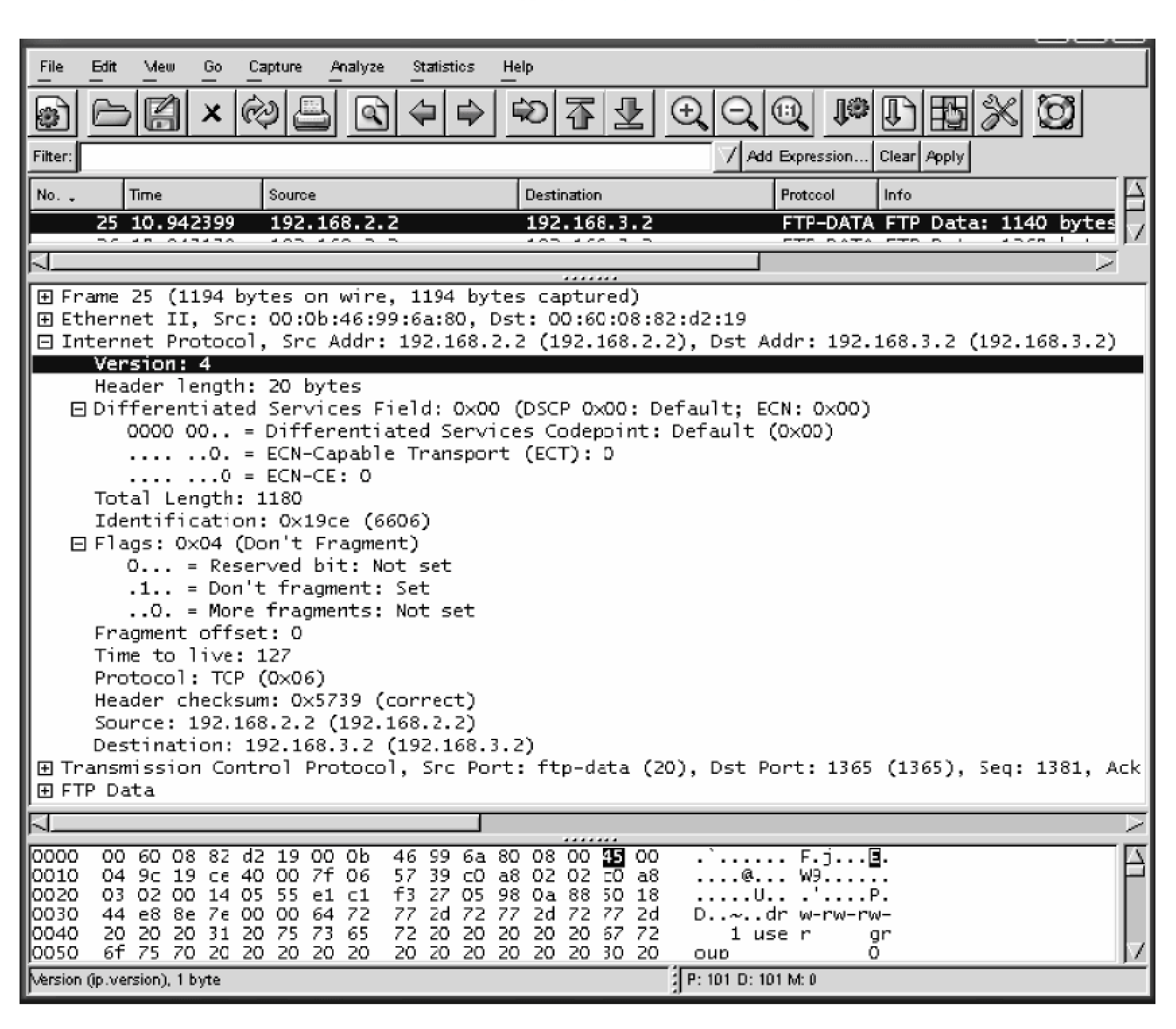



图　2-2

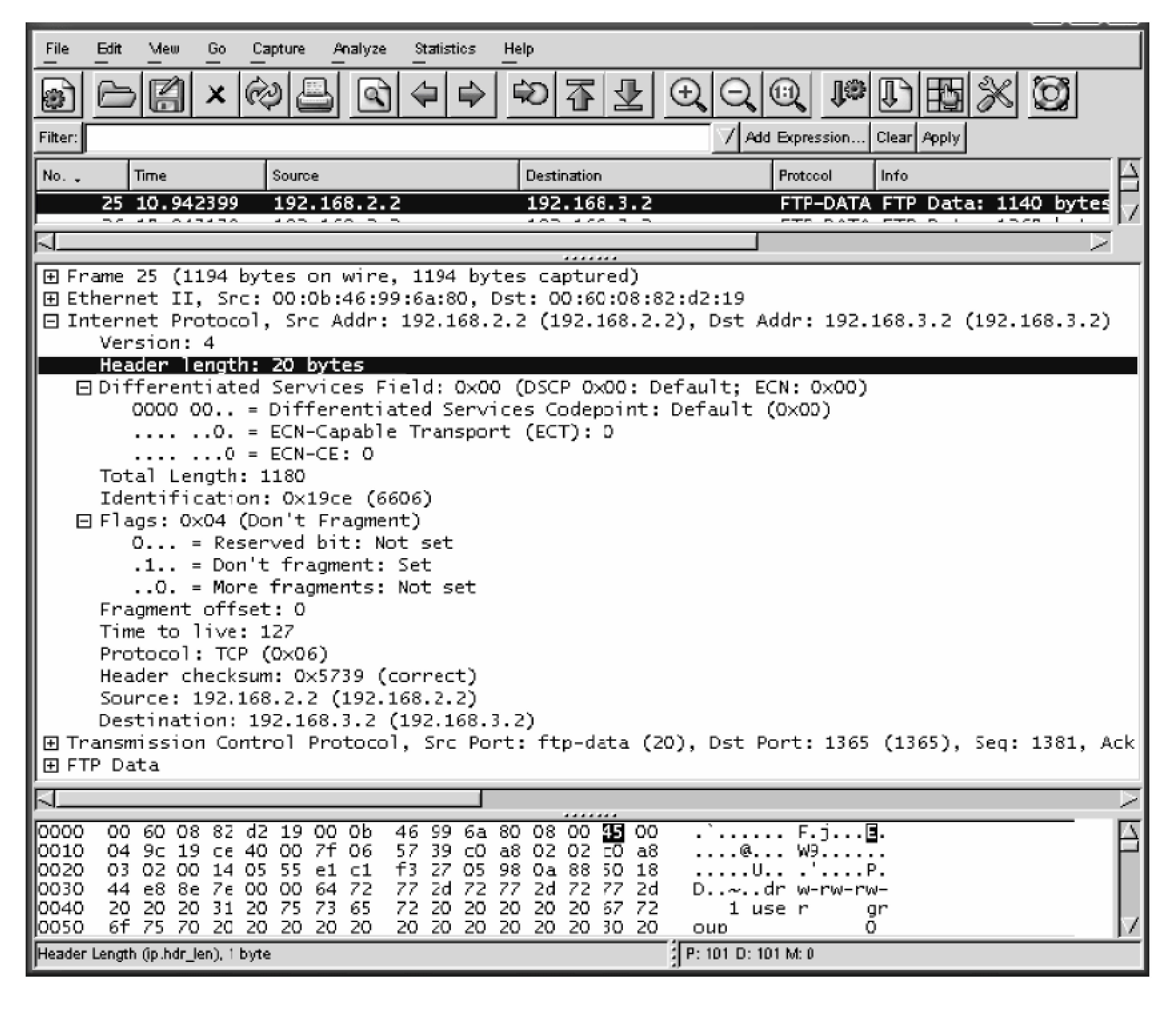



图 2-3

| | |
|---|---|
| Version：4 | 目前大部分 TCP/IP 产品使用 IPv4。 |
| Header Length：20 | IP 首部大小为 20 字节，从图 2-1 可知，IP 封包每行有 32 位，也就是 4 字节，那么 5 列就是 20 个字节。如果选项没有设定的话，那么包头的最短长度是 20 字节。 |

**2. Differentiated Services Field**

接下来的字段 Differentiated Services Field，如图 2-4 所示，它是为 IP 数据包提供优先级能力，这些优先级能力作用于能够利用它们的应用程序、主机和路由器上。通过适当的设置这些字段，一个应用程序可以请求它产生的数据包得到优先于其他等待通过的数据包的服务。

尽管该标志自 IPv4 首次发布以来就可以用了，但真正使用它的却只是少数应用程序。此外，只有少量的 IP 软件包和路由器支持它，这使得应用程序使用它没什么意义。

| | | |
|---|---|---|
| 000..... | Routine | 优先权要求，设为 0 为普通优先级，否则，数值越高越优先。 |
| ...0.... | Delay | 延迟要求，0 是普通值，1 为最小延迟。 |
| ....0... | Throughput | 通信量要求，0 为普通值，1 为最大吞吐量。 |
| .....0.. | Reliability | 可靠性要求，0 为普通值，1 为最高可靠性。 |
| ......00 | Not Used | 未使用。 |

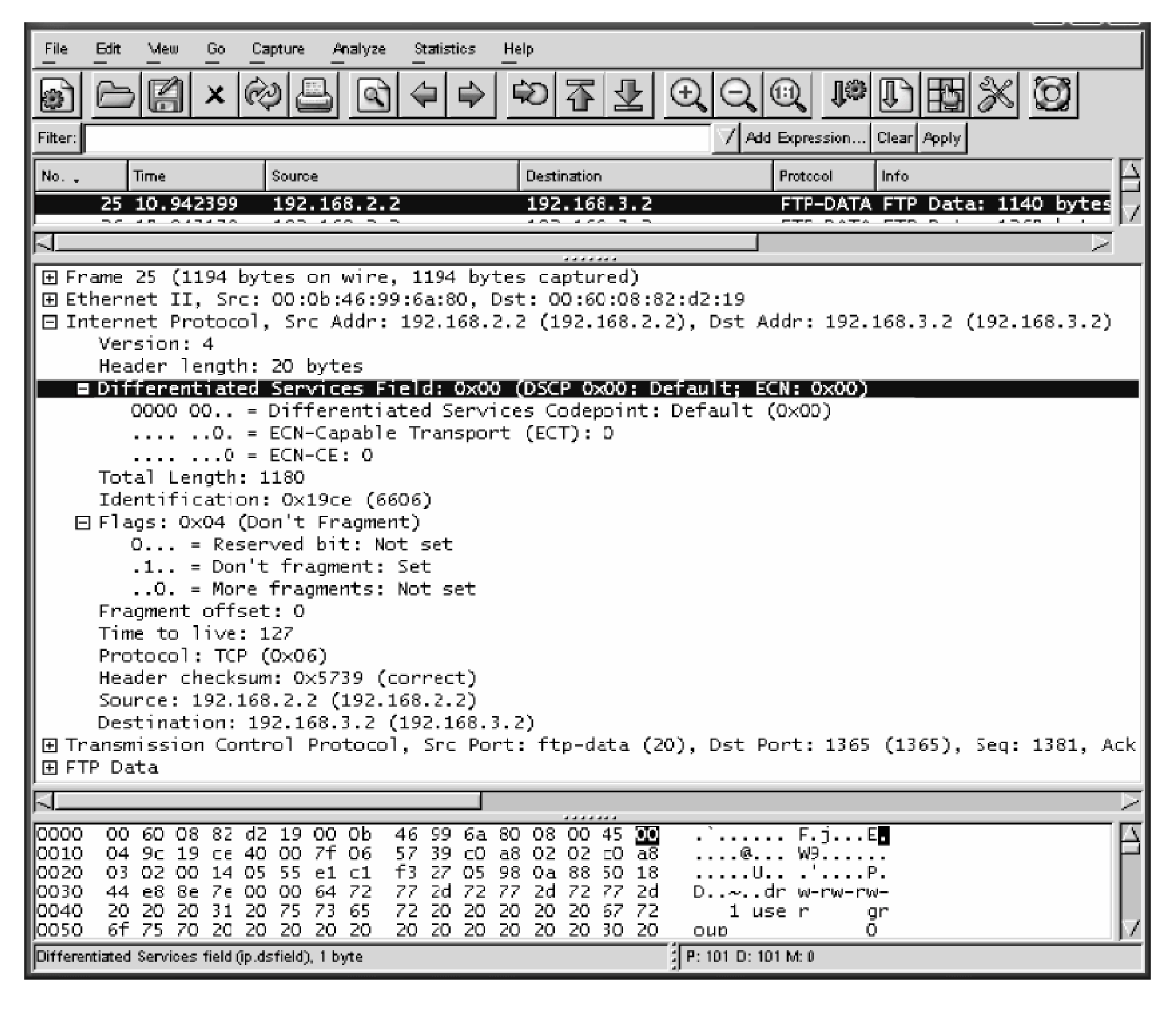



图　2-4

### 3. Total Length

说明整个 IP 数据包的大小，包括首部和数据部分，以字节表示。如图 2-5 所示，可以看到 Total Length：1180，同时注意到图最下方有个黑色加强的部分 04 9c，这个数字是 1180 的十六进制表示。

这个字段的主要目的是告诉系统该包的终止位置：total packet length－（减去）header length。

### 4. Identification

如图 2-6、图 2-7 所示，标识字段唯一地标识主机发送的每一份数据包。产生的每一个数据包都有 16 位的序列号，用来让发送系统和接收系统识别该数据包。通常每发送一份报文它的值就会加 1。

当要发送一个数据包分片的时候，会把这个字段的内容复制到每个片中，表示这些被分割的片属于同一个数据包。

图 2-6 和图 2-7 中两个数据包的序号分别为 6606 和 6607，在图 2-7 中，数据包的 Identification 字段的值是图 2-6 中的值加 1。

### 5. Flags

如图 2-8 所示，这个字段占有 3 位，它们的表示的意义如下：

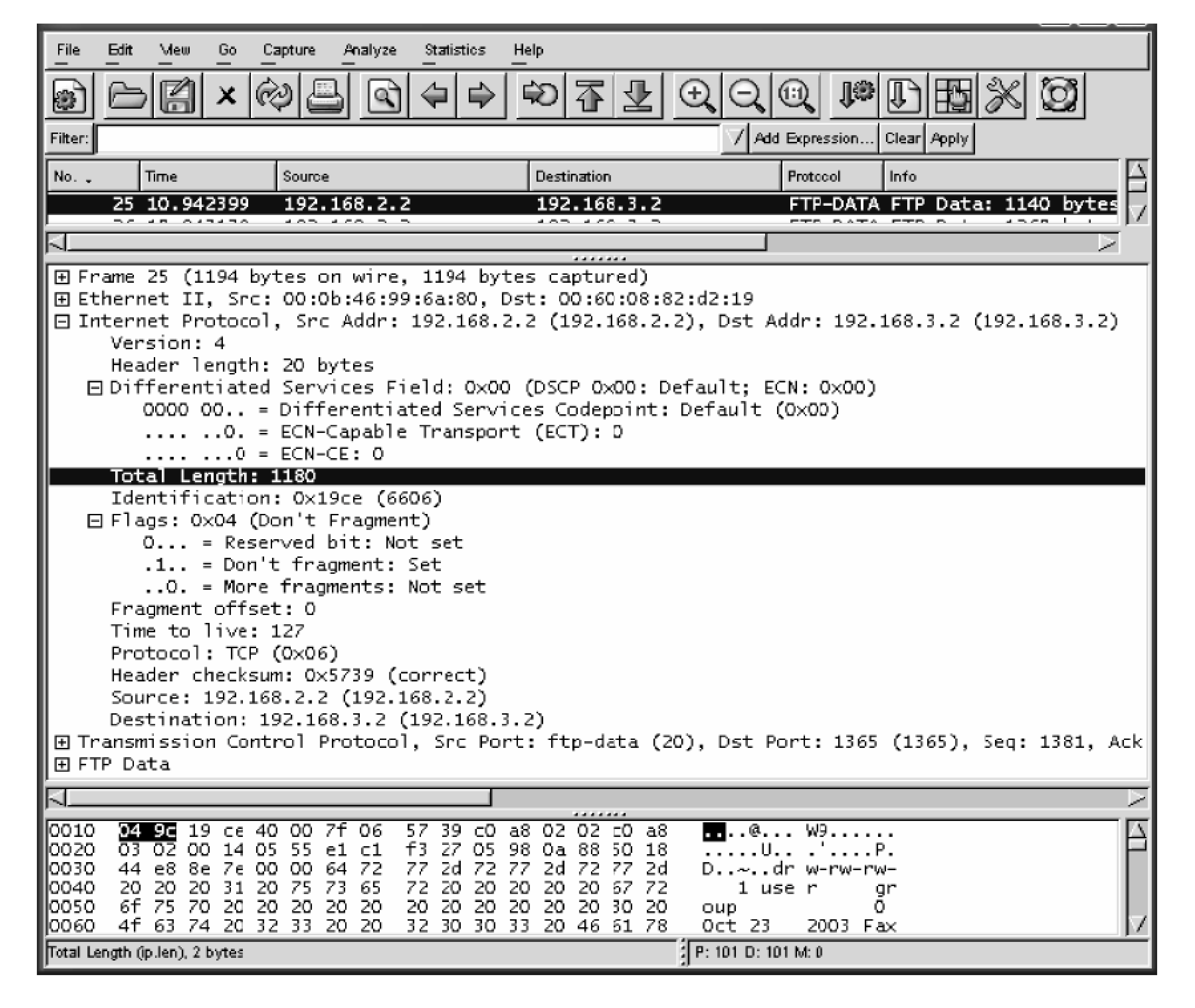

File Edit View Go Capture Analyze Statistics Help
Filter: Add Expression... Clear Apply
No. Time Source Destination Protocol Info
25 10.942399 192.168.2.2 192.168.3.2 FTP-DATA FTP Data: 1140 bytes
⊞ Frame 25 (1194 bytes on wire, 1194 bytes captured)
⊞ Ethernet II, Src: 00:0b:46:99:6a:80, Dst: 00:60:08:82:d2:19
⊟ Internet Protocol, Src Addr: 192.168.2.2 (192.168.2.2), Dst Addr: 192.168.3.2 (192.168.3.2)
Version: 4
Header length: 20 bytes
⊟ Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
0000 00.. = Differentiated Services Codepoint: Default (0x00)
.... ..0. = ECN-Capable Transport (ECT): 0
.... ...0 = ECN-CE: 0
Total Length: 1180
Identification: 0x19ce (6606)
⊟ Flags: 0x04 (Don't Fragment)
0... = Reserved bit: Not set
.1.. = Don't fragment: Set
..0. = More fragments: Not set
Fragment offset: 0
Time to live: 127
Protocol: TCP (0x06)
Header checksum: 0x5739 (correct)
Source: 192.168.2.2 (192.168.2.2)
Destination: 192.168.3.2 (192.168.3.2)
⊞ Transmission Control Protocol, Src Port: ftp-data (20), Dst Port: 1365 (1365), Seq: 1381, Ack
⊞ FTP Data
0010 04 9c 19 ce 40 00 7f 06 57 39 c0 a8 02 02 c0 a8 ....@... W9......
0020 03 02 00 14 05 55 e1 c1 f3 27 05 98 0a 88 50 18 .....U.. .'....P.
0030 44 e8 8e 7e 00 00 64 72 77 2d 72 77 2d 72 77 2d D..~..dr w-rw-rw-
0040 20 20 20 31 20 75 73 65 72 20 20 20 20 20 67 72 1 use r gr
0050 6f 75 70 20 20 20 20 20 20 20 20 20 20 20 30 20 oup 0
0060 4f 63 74 20 32 33 20 20 32 30 30 33 20 46 61 78 Oct 23 2003 Fax
Total Length (ip.len), 2 bytes
P: 101 D: 101 M: 0

图 2-5

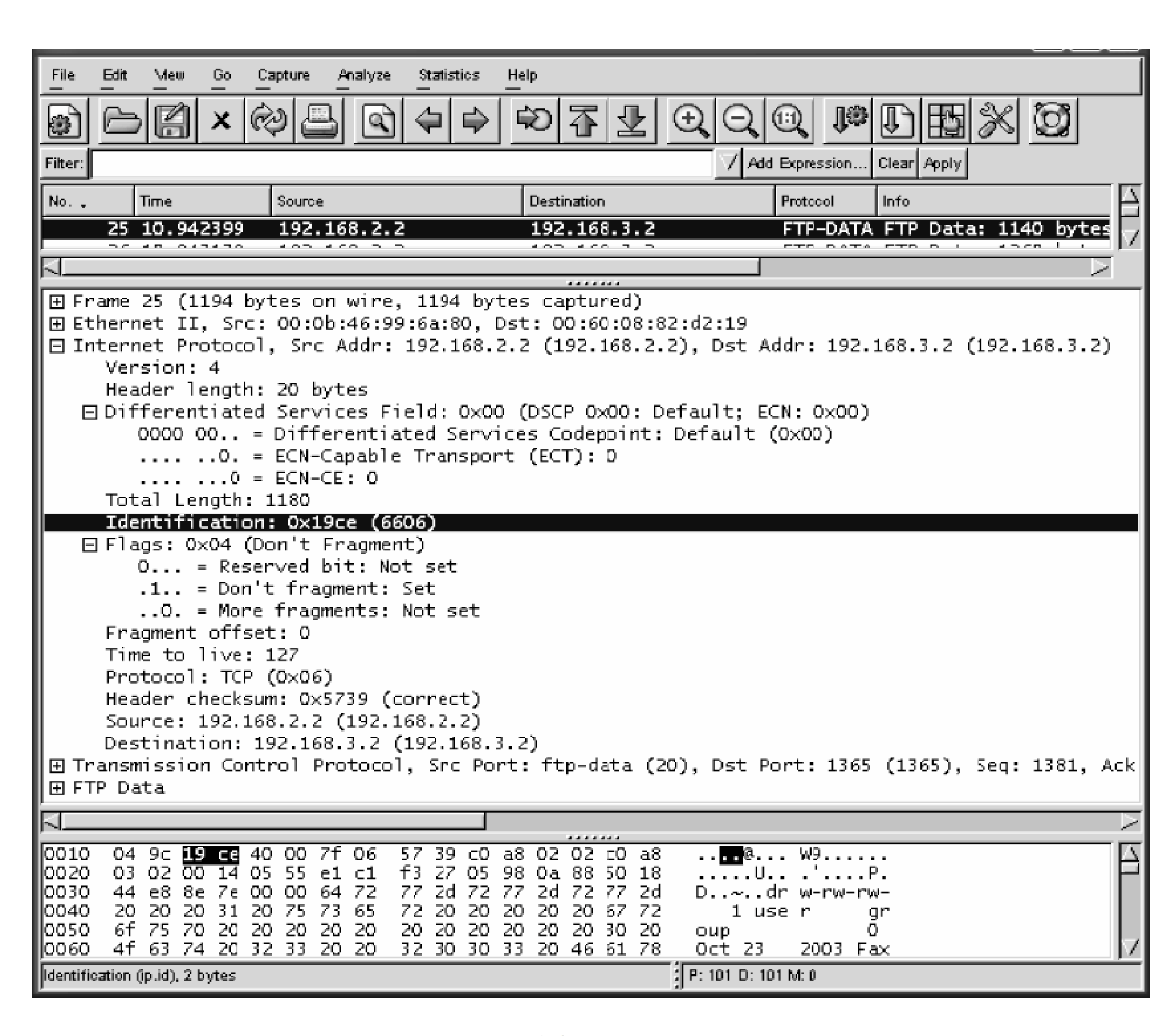

File Edit View Go Capture Analyze Statistics Help
Filter: Add Expression... Clear Apply
No. Time Source Destination Protocol Info
25 10.942399 192.168.2.2 192.168.3.2 FTP-DATA FTP Data: 1140 bytes
⊞ Frame 25 (1194 bytes on wire, 1194 bytes captured)
⊞ Ethernet II, Src: 00:0b:46:99:6a:80, Dst: 00:60:08:82:d2:19
⊟ Internet Protocol, Src Addr: 192.168.2.2 (192.168.2.2), Dst Addr: 192.168.3.2 (192.168.3.2)
Version: 4
Header length: 20 bytes
⊟ Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
0000 00.. = Differentiated Services Codepoint: Default (0x00)
.... ..0. = ECN-Capable Transport (ECT): 0
.... ...0 = ECN-CE: 0
Total Length: 1180
Identification: 0x19ce (6606)
⊟ Flags: 0x04 (Don't Fragment)
0... = Reserved bit: Not set
.1.. = Don't fragment: Set
..0. = More fragments: Not set
Fragment offset: 0
Time to live: 127
Protocol: TCP (0x06)
Header checksum: 0x5739 (correct)
Source: 192.168.2.2 (192.168.2.2)
Destination: 192.168.3.2 (192.168.3.2)
⊞ Transmission Control Protocol, Src Port: ftp-data (20), Dst Port: 1365 (1365), Seq: 1381, Ack
⊞ FTP Data
0010 04 9c 19 ce 40 00 7f 06 57 39 c0 a8 02 02 c0 a8 ....@... W9......
0020 03 02 00 14 05 55 e1 c1 f3 27 05 98 0a 88 50 18 .....U.. .'....P.
0030 44 e8 8e 7e 00 00 64 72 77 2d 72 77 2d 72 77 2d D..~..dr w-rw-rw-
0040 20 20 20 31 20 75 73 65 72 20 20 20 20 20 67 72 1 use r gr
0050 6f 75 70 20 20 20 20 20 20 20 20 20 20 20 30 20 oup 0
0060 4f 63 74 20 32 33 20 20 32 30 30 33 20 46 61 78 Oct 23 2003 Fax
Identification (ip.id), 2 bytes
P: 101 D: 101 M: 0

图 2-6

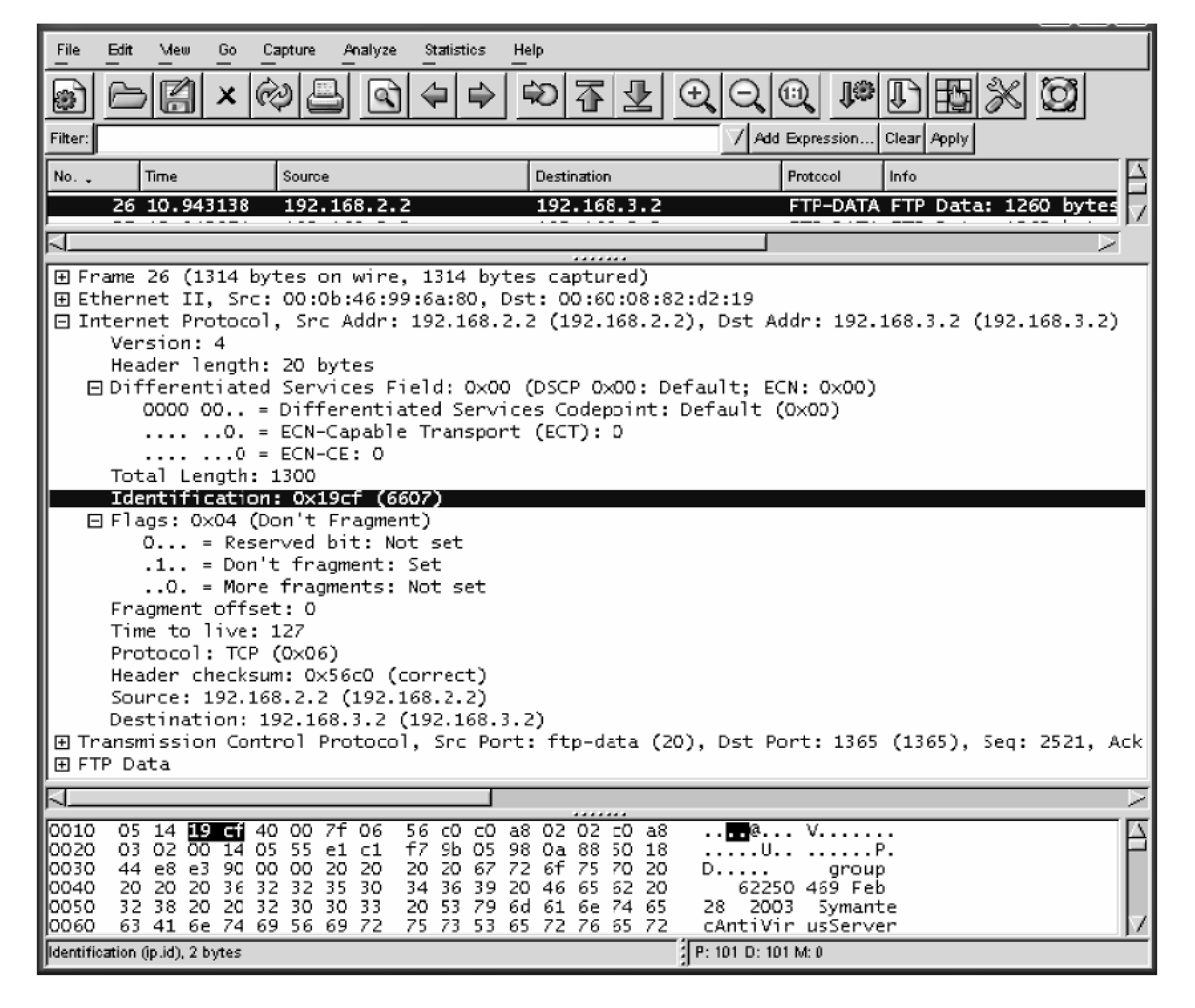

File Edit View Go Capture Analyze Statistics Help
Filter:
Add Expression... Clear Apply
No. Time Source Destination Protocol Info
26 10.943138 192.168.2.2 192.168.3.2 FTP-DATA FTP Data: 1260 bytes
Frame 26 (1314 bytes on wire, 1314 bytes captured)
Ethernet II, Src: 00:0b:46:99:6a:80, Dst: 00:60:08:82:d2:19
Internet Protocol, Src Addr: 192.168.2.2 (192.168.2.2), Dst Addr: 192.168.3.2 (192.168.3.2)
Version: 4
Header length: 20 bytes
Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
0000 00.. = Differentiated Services Codepoint: Default (0x00)
.... ..0. = ECN-Capable Transport (ECT): 0
.... ...0 = ECN-CE: 0
Total Length: 1300
Identification: 0x19cf (6607)
Flags: 0x04 (Don't Fragment)
0... = Reserved bit: Not set
.1.. = Don't fragment: Set
..0. = More fragments: Not set
Fragment offset: 0
Time to live: 127
Protocol: TCP (0x06)
Header checksum: 0x56c0 (correct)
Source: 192.168.2.2 (192.168.2.2)
Destination: 192.168.3.2 (192.168.3.2)
Transmission Control Protocol, Src Port: ftp-data (20), Dst Port: 1365 (1365), Seq: 2521, Ack
FTP Data
0010 05 14 19 cf 40 00 7f 06 56 c0 c0 a8 02 02 c0 a8
0020 03 02 00 14 05 55 e1 c1 f7 9b 05 98 0a 88 50 18
0030 44 e8 e3 9c 00 00 20 20 20 20 67 72 6f 75 70 20
0040 20 20 20 36 32 32 35 30 34 36 39 20 46 65 62 20
0050 32 38 20 20 32 30 30 33 20 53 79 6d 61 6e 74 65
0060 63 41 6e 74 69 56 69 72 75 73 53 65 72 76 65 72
Identification (ip.id), 2 bytes
P: 101 D: 101 M: 0

图　2-7

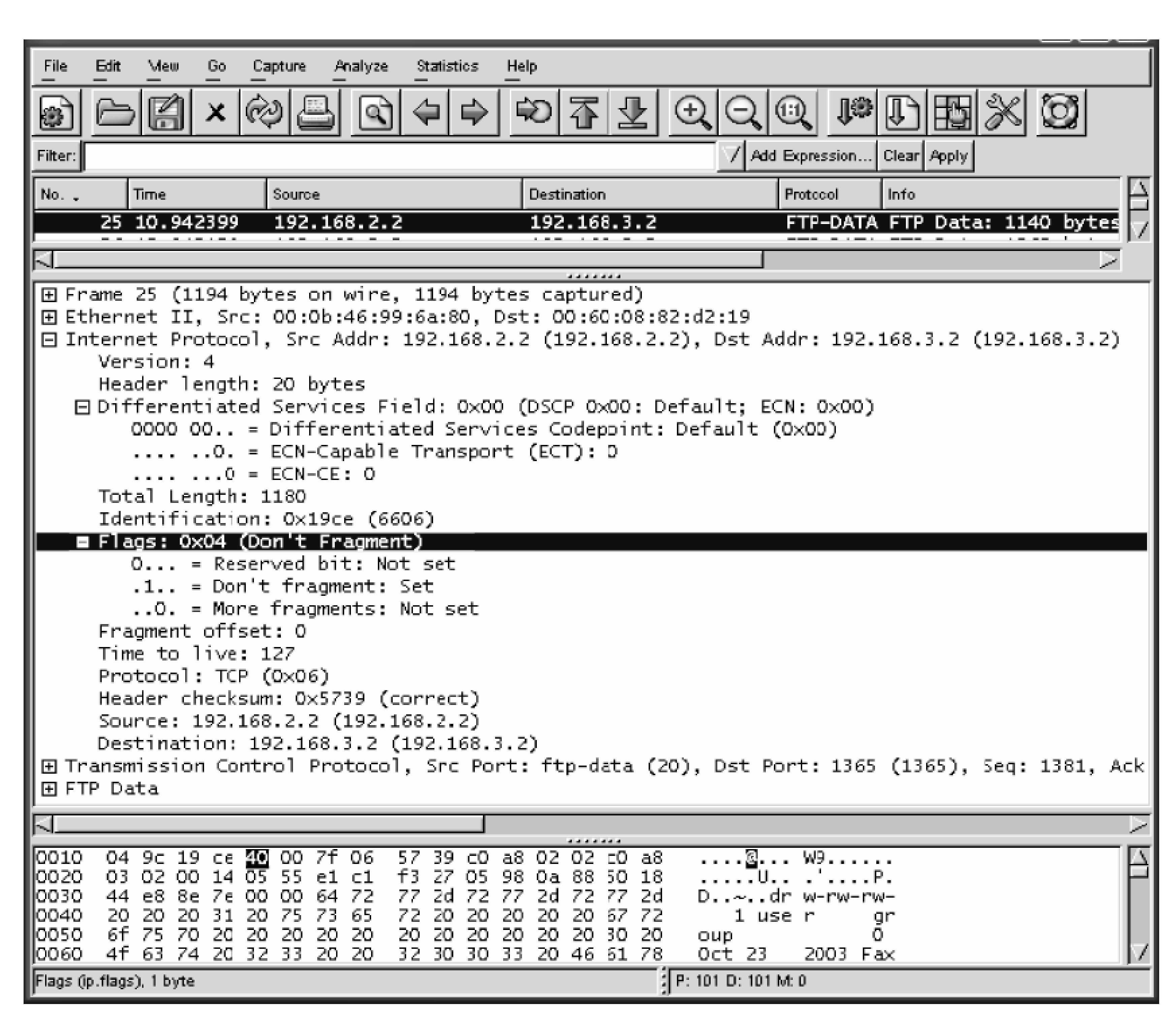

File Edit View Go Capture Analyze Statistics Help
Filter:
Add Expression... Clear Apply
No. Time Source Destination Protocol Info
25 10.942399 192.168.2.2 192.168.3.2 FTP-DATA FTP Data: 1140 bytes
Frame 25 (1194 bytes on wire, 1194 bytes captured)
Ethernet II, Src: 00:0b:46:99:6a:80, Dst: 00:60:08:82:d2:19
Internet Protocol, Src Addr: 192.168.2.2 (192.168.2.2), Dst Addr: 192.168.3.2 (192.168.3.2)
Version: 4
Header length: 20 bytes
Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
0000 00.. = Differentiated Services Codepoint: Default (0x00)
.... ..0. = ECN-Capable Transport (ECT): 0
.... ...0 = ECN-CE: 0
Total Length: 1180
Identification: 0x19ce (6606)
Flags: 0x04 (Don't Fragment)
0... = Reserved bit: Not set
.1.. = Don't fragment: Set
..0. = More fragments: Not set
Fragment offset: 0
Time to live: 127
Protocol: TCP (0x06)
Header checksum: 0x5739 (correct)
Source: 192.168.2.2 (192.168.2.2)
Destination: 192.168.3.2 (192.168.3.2)
Transmission Control Protocol, Src Port: ftp-data (20), Dst Port: 1365 (1365), Seq: 1381, Ack
FTP Data
0010 04 9c 19 ce 40 00 7f 06 57 39 c0 a8 02 02 c0 a8
0020 03 02 00 14 05 55 e1 c1 f3 27 05 98 0a 88 50 18
0030 44 e8 8e 7e 00 00 64 72 77 2d 72 77 2d 72 77 2d
0040 20 20 20 31 20 75 73 65 72 20 20 20 20 20 67 72
0050 6f 75 70 20 20 20 20 20 20 20 20 20 20 20 30 20
0060 4f 63 74 20 32 33 20 20 32 30 30 33 20 46 61 78
Flags (ip.flags), 1 byte
P: 101 D: 101 M: 0

图　2-8

Reserved bit：目前没有使用。

don't fragment：用于说明某一 IP 路由器是否可以分片这个 IP 包。为 1 时表示不能被分割。

more fragments：用来说明源数据包是否还有其他片段。为 1 时表示其后还有被分割的包。

### 6. Fragment Offset

如图 2-9 所示，这个字段是偏移定位，当一个大的数据包在经过一些传输单元（MTU）较小的路径时，会被分割成片 fragment 进行传送（这个切片和传送层的打包有所不同，它是由链路层决定的），由于链路情况和其他因素的影响，数据到达时有可能会失序，数据到达时的顺序不一定就是当初分割数据包的顺序，所以当分片的时候，每个片做好偏移定位标记，便于在重组的时候对号入座。

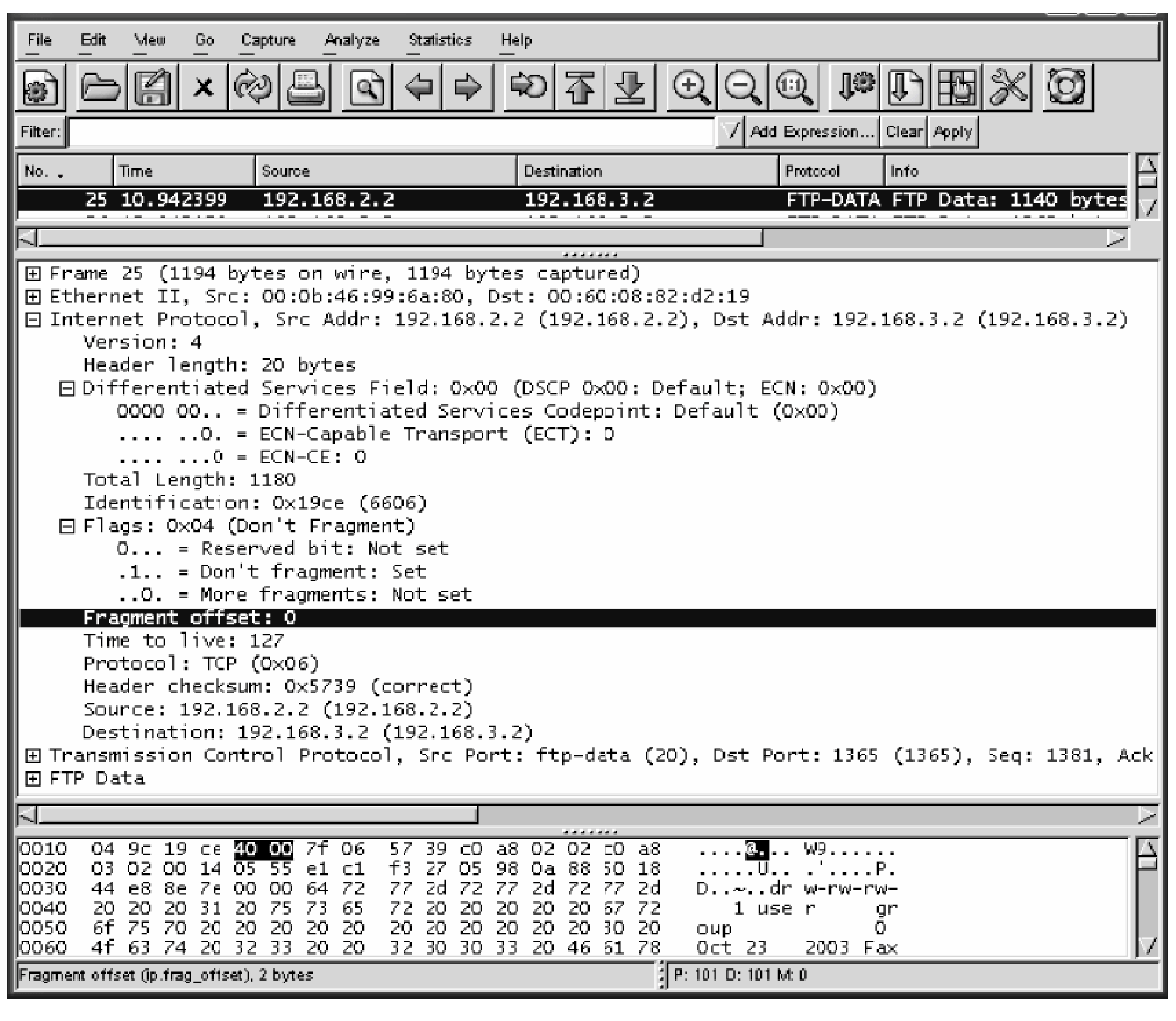



图 2-9

fragmentation offset 设为 0，表示数据包没有被分片。

因为没有提供"片段的序列号"字段，目的系统地端把这个字段和 total packet length 和 don't fragment 标志一起使用。

图 2-9 和图 2-8 的 flags 放在一起看，不难看出"40 00"表示该片段不可分，这是第一个也是最后一个。

### 7. Time-to-Live

如图 2-10 所示，生存时间（TTL），在许多网络协议中都会碰到，它指定某个数据包在不能发送并被丢弃之前可以经过的最大跳数。当某个源端产生一个 IP 数据包时，它就在

Time-to-live 字段显示一个 1～255 之间的值。每当该包经过路由器发送一次，这个值就减去 1。如果这个值在到达最终目的系统之前减到 0，该包就被认为是不可发送的并被立刻丢弃。

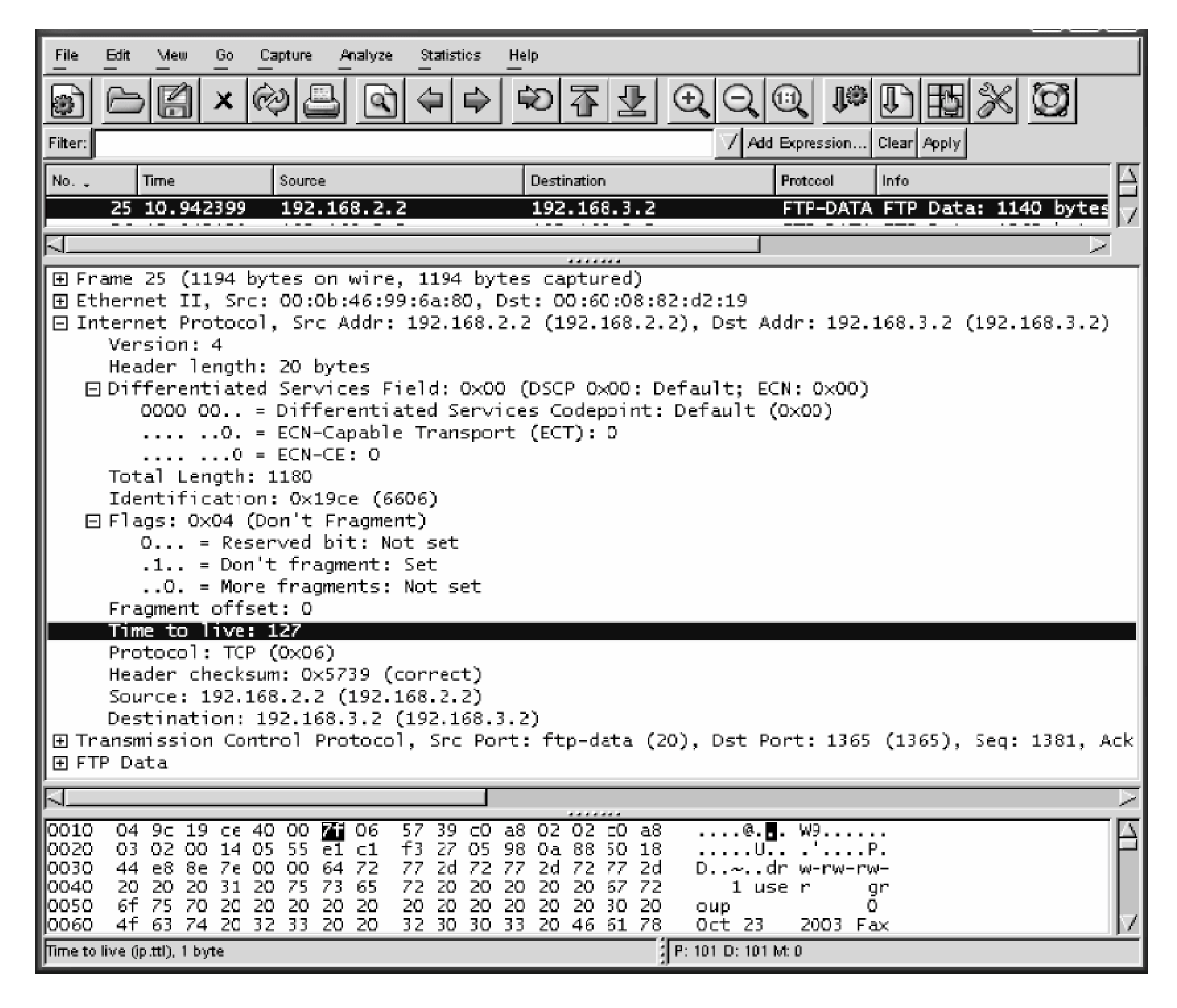



图　2-10

**8．Protocol**

这个字段用来识别嵌入到 IP 数据包中的上层协议的类型：值为 1 表示 ICMP，值为 2 表示 IGMP，值为 6 表示 TCP，值为 17 表示 UDP。

图 2-11 中数值为 6，表示 TCP 协议。

**9．Header Checksum**

如图 2-12 所示，用以储存该 IP 首部的校验和，使得中间设备能够验证首部的内容并检验出可能的数据损坏。

校验和仅仅应用于 IP 首部的值，而不应用于整个 IP 数据包。

**10．Source Address**

如图 2-13 所示，标识该数据包的源发送者，也就是源端系统使用的 32 位 IP 地址。

**11．Destination Address**

如图 2-14 所示，标识数据包的最终目的地的 IP 地址。

这个字段说明的是该数据包的最终目的地址，为了把数据包发送到最终目的地，目的地的 IP 地址必须在首部中提供，并且必须总是保留在首部中。

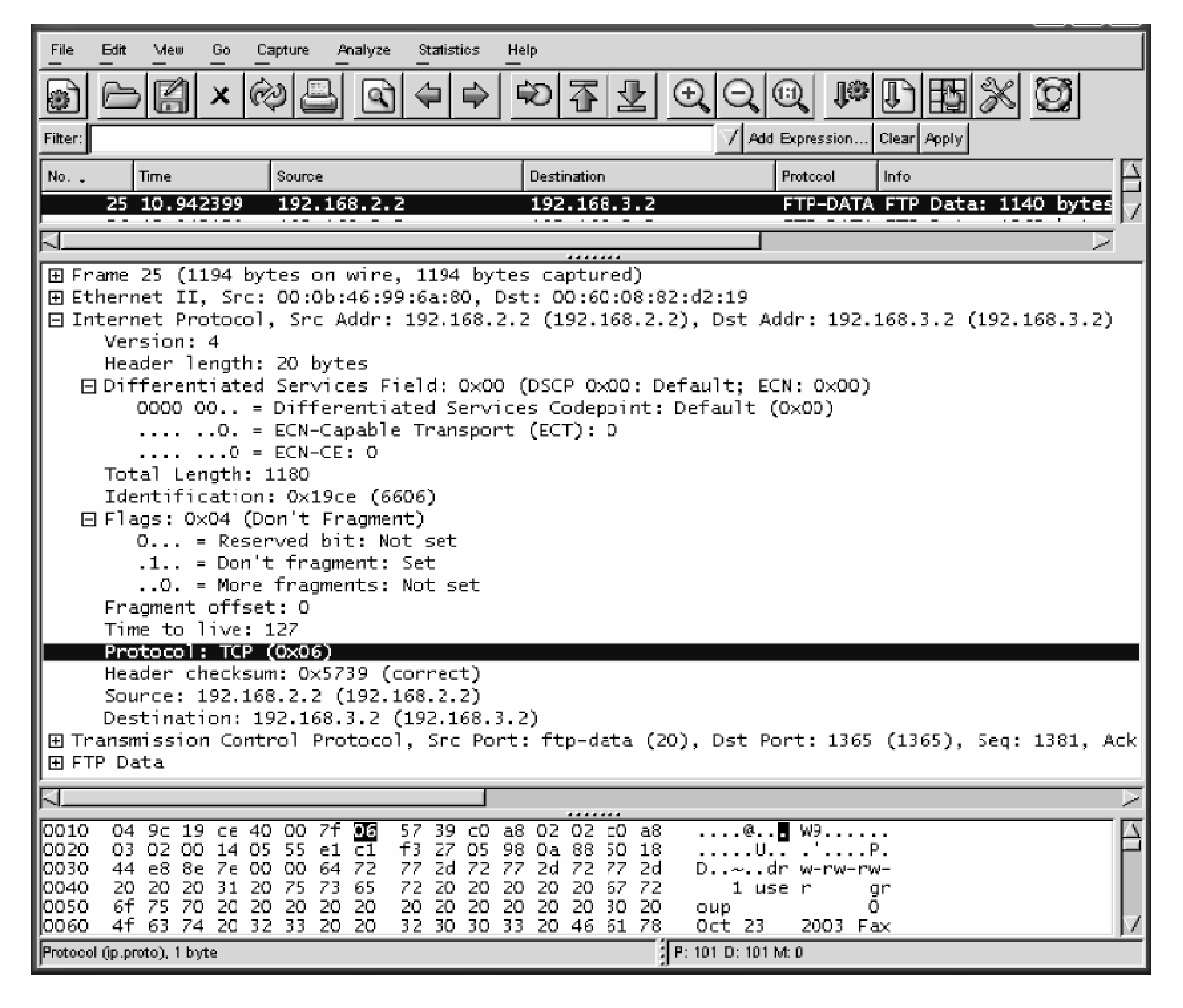

File Edit View Go Capture Analyze Statistics Help
Filter:
Add Expression... Clear Apply
No. Time Source Destination Protocol Info
25 10.942399 192.168.2.2 192.168.3.2 FTP-DATA FTP Data: 1140 bytes
Frame 25 (1194 bytes on wire, 1194 bytes captured)
Ethernet II, Src: 00:0b:46:99:6a:80, Dst: 00:60:08:82:d2:19
Internet Protocol, Src Addr: 192.168.2.2 (192.168.2.2), Dst Addr: 192.168.3.2 (192.168.3.2)
Version: 4
Header length: 20 bytes
Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
0000 00.. = Differentiated Services Codepoint: Default (0x00)
.... ..0. = ECN-Capable Transport (ECT): 0
.... ...0 = ECN-CE: 0
Total Length: 1180
Identification: 0x19ce (6606)
Flags: 0x04 (Don't Fragment)
0... = Reserved bit: Not set
.1.. = Don't fragment: Set
..0. = More fragments: Not set
Fragment offset: 0
Time to live: 127
Protocol: TCP (0x06)
Header checksum: 0x5739 (correct)
Source: 192.168.2.2 (192.168.2.2)
Destination: 192.168.3.2 (192.168.3.2)
Transmission Control Protocol, Src Port: ftp-data (20), Dst Port: 1365 (1365), Seq: 1381, Ack
FTP Data
0010 04 9c 19 ce 40 00 7f 06 57 39 c0 a8 02 02 c0 a8 ....@... W9......
0020 03 02 00 14 05 55 e1 c1 f3 27 05 98 0a 88 50 18 .....U.. .'....P.
0030 44 e8 8e 7e 00 00 64 72 77 2d 72 77 2d 72 77 2d D..~..dr w-rw-rw-
0040 20 20 20 31 20 75 73 65 72 20 20 20 20 20 67 72 1 use r gr
0050 6f 75 70 20 20 20 20 20 20 20 20 20 20 20 30 20 oup 0
0060 4f 63 74 20 32 33 20 20 32 30 30 33 20 46 61 78 Oct 23 2003 Fax
Protocol (ip.proto), 1 byte
P: 101 D: 101 M: 0

图 2-11

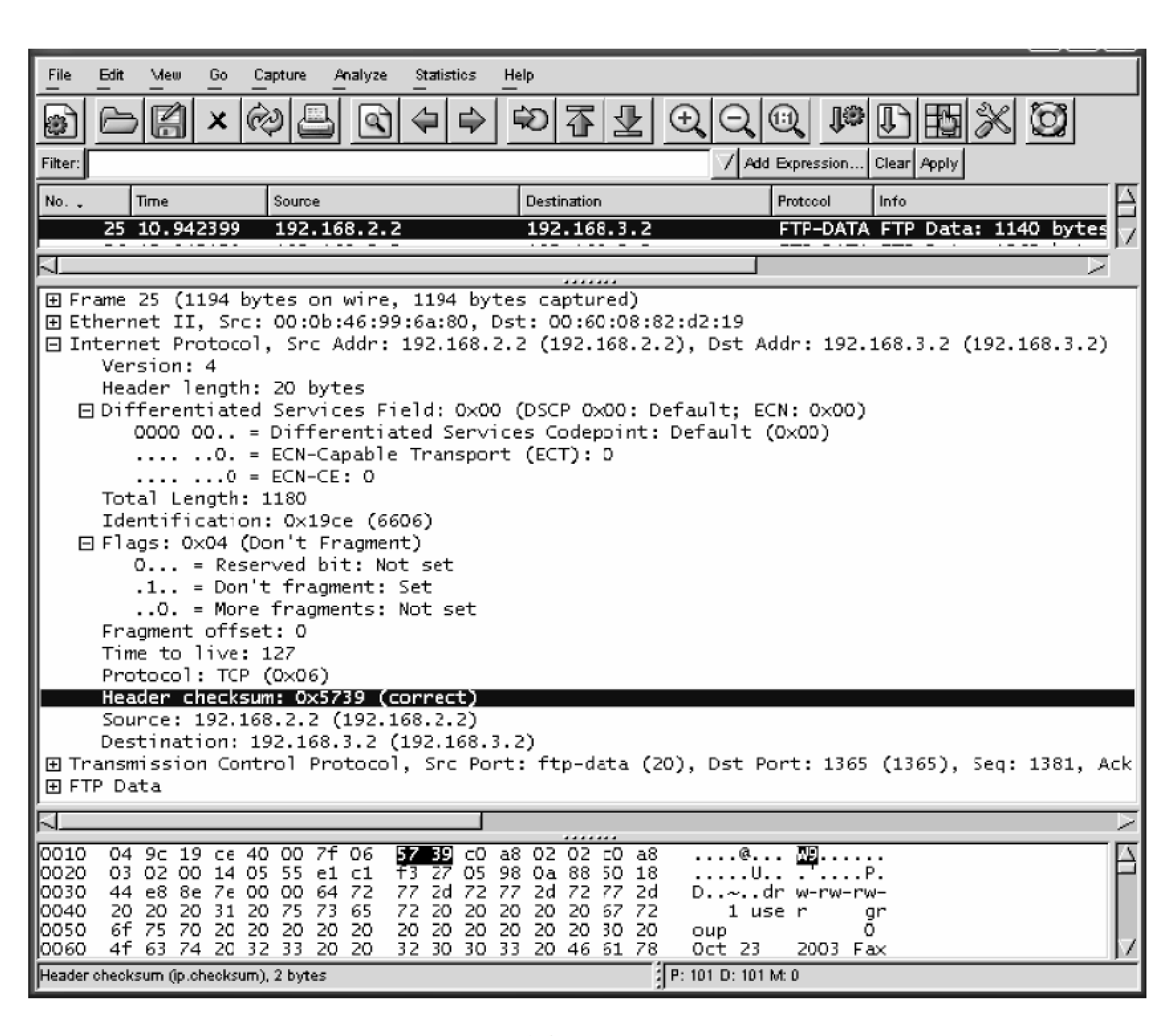

File Edit View Go Capture Analyze Statistics Help
Filter:
Add Expression... Clear Apply
No. Time Source Destination Protocol Info
25 10.942399 192.168.2.2 192.168.3.2 FTP-DATA FTP Data: 1140 bytes
Frame 25 (1194 bytes on wire, 1194 bytes captured)
Ethernet II, Src: 00:0b:46:99:6a:80, Dst: 00:60:08:82:d2:19
Internet Protocol, Src Addr: 192.168.2.2 (192.168.2.2), Dst Addr: 192.168.3.2 (192.168.3.2)
Version: 4
Header length: 20 bytes
Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
0000 00.. = Differentiated Services Codepoint: Default (0x00)
.... ..0. = ECN-Capable Transport (ECT): 0
.... ...0 = ECN-CE: 0
Total Length: 1180
Identification: 0x19ce (6606)
Flags: 0x04 (Don't Fragment)
0... = Reserved bit: Not set
.1.. = Don't fragment: Set
..0. = More fragments: Not set
Fragment offset: 0
Time to live: 127
Protocol: TCP (0x06)
Header checksum: 0x5739 (correct)
Source: 192.168.2.2 (192.168.2.2)
Destination: 192.168.3.2 (192.168.3.2)
Transmission Control Protocol, Src Port: ftp-data (20), Dst Port: 1365 (1365), Seq: 1381, Ack
FTP Data
0010 04 9c 19 ce 40 00 7f 06 57 39 c0 a8 02 02 c0 a8 ....@... W9......
0020 03 02 00 14 05 55 e1 c1 f3 27 05 98 0a 88 50 18 .....U.. .'....P.
0030 44 e8 8e 7e 00 00 64 72 77 2d 72 77 2d 72 77 2d D..~..dr w-rw-rw-
0040 20 20 20 31 20 75 73 65 72 20 20 20 20 20 67 72 1 use r gr
0050 6f 75 70 20 20 20 20 20 20 20 20 20 20 20 30 20 oup 0
0060 4f 63 74 20 32 33 20 20 32 30 30 33 20 46 61 78 Oct 23 2003 Fax
Header checksum (ip.checksum), 2 bytes
P: 101 D: 101 M: 0

图 2-12

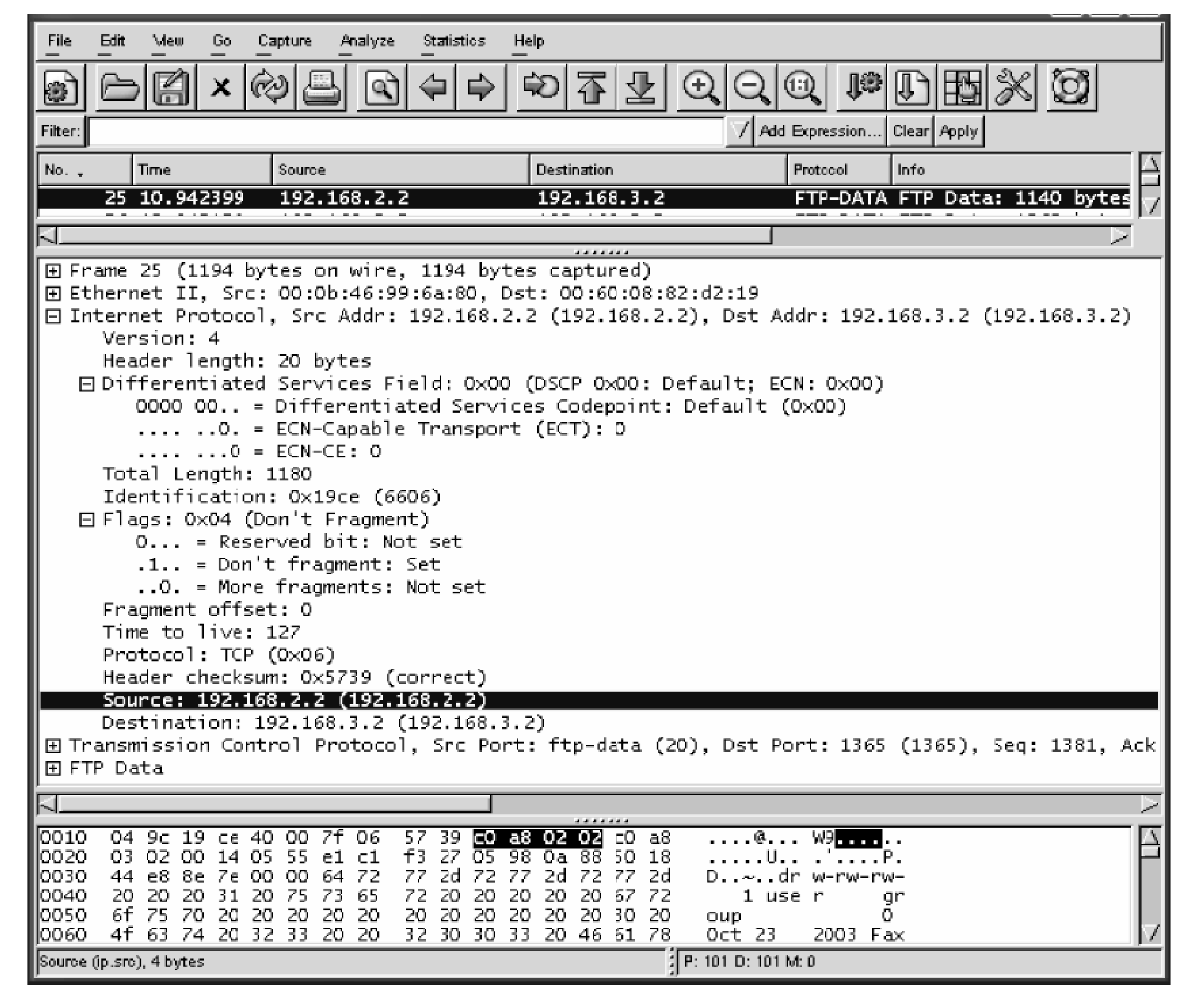

File Edit View Go Capture Analyze Statistics Help
Filter:
Add Expression... Clear Apply
No. Time Source Destination Protocol Info
25 10.942399 192.168.2.2 192.168.3.2 FTP-DATA FTP Data: 1140 bytes
Frame 25 (1194 bytes on wire, 1194 bytes captured)
Ethernet II, Src: 00:0b:46:99:6a:80, Dst: 00:60:08:82:d2:19
Internet Protocol, Src Addr: 192.168.2.2 (192.168.2.2), Dst Addr: 192.168.3.2 (192.168.3.2)
Version: 4
Header length: 20 bytes
Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
0000 00.. = Differentiated Services Codepoint: Default (0x00)
.... ..0. = ECN-Capable Transport (ECT): 0
.... ...0 = ECN-CE: 0
Total Length: 1180
Identification: 0x19ce (6606)
Flags: 0x04 (Don't Fragment)
0... = Reserved bit: Not set
.1.. = Don't fragment: Set
..0. = More fragments: Not set
Fragment offset: 0
Time to live: 127
Protocol: TCP (0x06)
Header checksum: 0x5739 (correct)
Source: 192.168.2.2 (192.168.2.2)
Destination: 192.168.3.2 (192.168.3.2)
Transmission Control Protocol, Src Port: ftp-data (20), Dst Port: 1365 (1365), Seq: 1381, Ack
FTP Data
0010 04 9c 19 ce 40 00 7f 06 57 39 c0 a8 02 02 c0 a8 ....@... W9......
0020 03 02 00 14 05 55 e1 c1 f3 27 05 98 0a 88 50 18 .....U.. .'....P.
0030 44 e8 8e 7e 00 00 64 72 77 2d 72 77 2d 72 77 2d D..~..dr w-rw-rw-
0040 20 20 20 31 20 75 73 65 72 20 20 20 20 20 67 72 1 use r gr
0050 6f 75 70 20 20 20 20 20 20 20 20 20 20 20 30 20 oup 0
0060 4f 63 74 20 32 33 20 20 32 30 30 33 20 46 61 78 Oct 23 2003 Fax
Source (ip.src), 4 bytes
P: 101 D: 101 M: 0

图　2-13

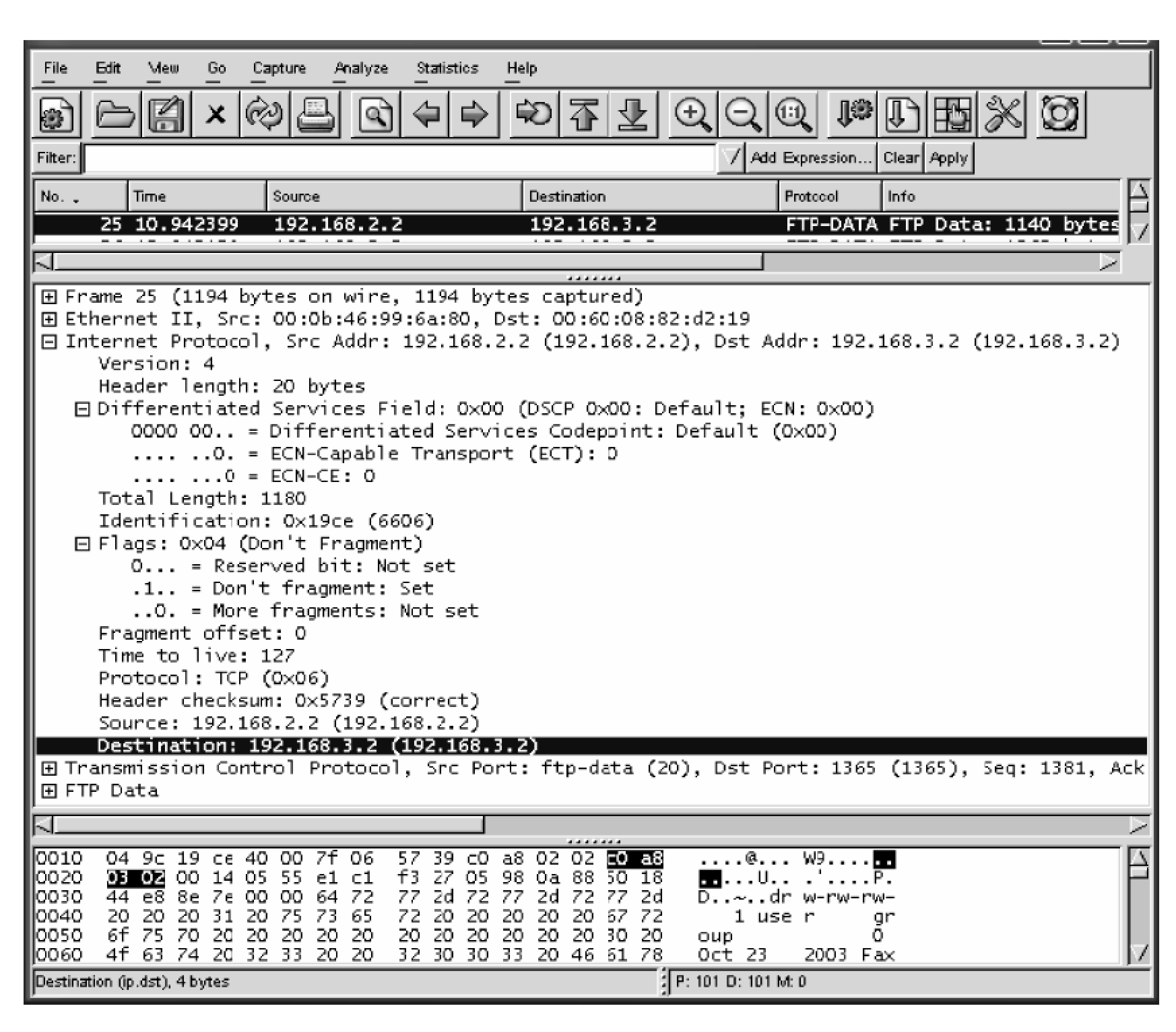

File Edit View Go Capture Analyze Statistics Help
Filter:
Add Expression... Clear Apply
No. Time Source Destination Protocol Info
25 10.942399 192.168.2.2 192.168.3.2 FTP-DATA FTP Data: 1140 bytes
Frame 25 (1194 bytes on wire, 1194 bytes captured)
Ethernet II, Src: 00:0b:46:99:6a:80, Dst: 00:60:08:82:d2:19
Internet Protocol, Src Addr: 192.168.2.2 (192.168.2.2), Dst Addr: 192.168.3.2 (192.168.3.2)
Version: 4
Header length: 20 bytes
Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
0000 00.. = Differentiated Services Codepoint: Default (0x00)
.... ..0. = ECN-Capable Transport (ECT): 0
.... ...0 = ECN-CE: 0
Total Length: 1180
Identification: 0x19ce (6606)
Flags: 0x04 (Don't Fragment)
0... = Reserved bit: Not set
.1.. = Don't fragment: Set
..0. = More fragments: Not set
Fragment offset: 0
Time to live: 127
Protocol: TCP (0x06)
Header checksum: 0x5739 (correct)
Source: 192.168.2.2 (192.168.2.2)
Destination: 192.168.3.2 (192.168.3.2)
Transmission Control Protocol, Src Port: ftp-data (20), Dst Port: 1365 (1365), Seq: 1381, Ack
FTP Data
0010 04 9c 19 ce 40 00 7f 06 57 39 c0 a8 02 02 c0 a8 ....@... W9......
0020 03 02 00 14 05 55 e1 c1 f3 27 05 98 0a 88 50 18 .....U.. .'....P.
0030 44 e8 8e 7e 00 00 64 72 77 2d 72 77 2d 72 77 2d D..~..dr w-rw-rw-
0040 20 20 20 31 20 75 73 65 72 20 20 20 20 20 67 72 1 use r gr
0050 6f 75 70 20 20 20 20 20 20 20 20 20 20 20 30 20 oup 0
0060 4f 63 74 20 32 33 20 20 32 30 30 33 20 46 61 78 Oct 23 2003 Fax
Destination (ip.dst), 4 bytes
P: 101 D: 101 M: 0

图　2-14

CHAPTER 3

# 第 3 章 TCP 和 UDP 协议

应用程序用来在 IP 网络上相互之间传输的标准传输协议有两个。一个是用户数据包协议(UDP),它提供的服务轻便但不可靠;另外一个是传输控制协议(TCP),它提供的是可靠的、可控制的传输服务。大部分 Internet 应用程序都使用 TCP,因为它的嵌入可靠性和流控制服务可确保数据不会丢失和被破坏。

## 3.1 TCP 协议

TCP 是目前 Internet 上使用的最重要的协议。IP 在 Internet 上发送数据包,TCP 确保了 IP 数据包中的数据的正确性,没有 TCP 可靠性服务,Internet 即使能够工作,也不可能工作得像现在这么好。

RFC 793 规定 TCP 的 Protocol ID 为 6。当系统收到标注了包含 Protocol 6 的数据包后,它就把该数据包的内容发送到 TCP 作进一步处理。

### 3.1.1 TCP 所提供的服务及 TCP 数据包结构

TCP 提供的服务大致可以分为与连接相关的服务和与数据传输相关的服务。

当数据从一台计算机传输到另一台计算机时,就需要 TCP 在源和目标计算机之间建立一条传输通道,用来实现两台计算机之间的数据交换。首先,发送端计算机上,使用 TCP 协议的应用程序在两台计算机开始通信时,通知它的操作系统启动与另一台计算机间的通信,并且告知接收端,想和它建立连接,此过程为主动打开。接收计算机上使用 TCP 协议的应用程序通知本地操作系统:自己和其他计算机的通信即将开始,此过程为被动打开。发送端和接收端的操作系统都分配一个定义好的信道,把这个信道称为端口。在两台计算机之间正在被调用的应用程序通过使用端口进行通信和数据交换。当发送和接收端完成了启动通信的过程并确认了所建立的连接后,就可以开始数据交换了。图 3-1 是 TCP 报头格式。

| 16位源端口号 | | | 16位目的端口号 |
|---|---|---|---|
| 32位序号 | | | |
| 32位确认序号 | | | |
| 4位首部长度 | 保留(6位) | 标志(6位) | 16位窗口大小 |
| 16位校验和 | | | 16位紧急指针 |
| 选项+padding | | | |
| 数据 | | | |

(前 20 个字节)

图 3-1

**1. 源端口和目标端口(Source Port/Destination Port,16 位)**

在每一个 TCP 的头部中都包含有源端口号和目的端口号,用于定位源端的应用进程和目的端的应用进程。在前面的章节中已讨论过通过 IP 地址可以唯一地在网络上确定一台主机;而通常每一台机器上都会有多个进程在同时运行,都可能向网络的服务器提出服务的请求或向网络中的其他客户提供服务,这时通过端口号来区分不同的应用进程。例如当某一台客户端的应用程序要向某一服务器提出 WWW 服务的请求时,用目的端口号 80 来向收到请求的服务器表明它所请求的服务为 WWW 服务;如果同时该机器的另一应用程序向服务器提出 FTP 服务的请求时,则目的端口号 21 表示它所需要的服务类型为 FTP。这样就很好地保证了在多个应用同时需要进行数据传输时传输的数据不至于混淆。

**2. 序列号和确认号(Sequence Number/ Acknowledgement,32 位)**

序列号和确认号是 TCP 实现可靠连接的关键。当建立一个 TCP 连接时,发送方主机发出一个随机的初始化序列号给接收方,接收方将其加 1 后送回发送方,这意味着发送方可以发送下一个字节了。一旦数据开始传送,序列号和确认号将跟踪已发送了的那些数据。因为每个域都是 32 位,总共可以有 $2^{32}$ 个值,所以每个域的范围是:0～4 294 967 295,当超过上限时回到 0。

**3. 首部长度(4 位)**

以字节为单位表示 TCP 头的大小。头长度随可变长度选项字段而改变,通过这个字段同时可以判断该 TCP 数据段的开始位置和结束位置。

**4. 标志(6 位)**

标志字段部分包含 6 个标志位,它说明了其他字段包含有意义的数据或说明某种控制功能。具体每个标志位的意义后面会讲到。

**5. 窗口(16 位)**

此字段告诉接收这个 TCP 报文的接收端自己还可以接收多少数据字节。它大致对应于滑动窗口协议的窗口尺寸。反过来,接收 TCP 报文的接收端可以使用此字段来改变发送端的窗口的大小。窗口最大为 65 535 字节。

#### 6. 校验和(16 位)

用于传输层差错检测。校验和算法将 TCP 段的内容转换为一系列 16 位的整数，并将它们相加。接收方根据校验和判断传输是否正确。

#### 7. 紧急指针(Urgent Pointer，16 位)

只有当 URG 标志置 1 时才有效。紧急指针是一个正的偏移量，和序号字段中的值相加表示紧急数据最后一个字节的序号。TCP 为了提高效率，数据不会直接发送到对方，一般都先放在数据缓冲区中，等到数据积累到一定的大小才会一起发送。紧急指针所指的一段数据不必等待缓冲数据的积累，直接发送到对方。

#### 8. 选项+padding(可变长度)

选项字段用于确定主机可以接收数据段的最大尺寸。通常在建立初始连接时规定此值。这是一个重要选项，因为 TCP 可能连接了两台数据处理能力相差很大的计算机，为了防止数据拥塞，要求每台计算机都了解另一台的所有限制(如缓冲区大小)。例如，一台服务器一次发送太大的数据段将导致 PC 机无法处理，因此 PC 会在建立 TCP 连接时确定它可接收数据的最大段尺寸。

TCP 首部最小为 20 字节。如果定义了一些选项，首部的大小就增加(最大为 60 字节)。RFC 793 规定首部必须可以被 32 位整除，所以如果定义了一个选项，但该选项只用了 16 位，那么必须使用 padding 字段补充 16 位，padding 字段是由 0 组成的字段。

#### 9. 数据(可变大小)

用户提供的数据。

### 3.1.2 TCP 数据传输原理

当 TCP 连接完成进行数据传输时，有可能出现一系列复杂的情况，比如网络拥塞、数据丢失、数据传输效率低下等。TCP 有一系列机制来保证数据传输提供和保证可靠的、安全的、有效的、高效率的数据传输，下面进行介绍。

#### 1. MSS(最大报文段长度)和 MTU(最大传输单元)大小

在 TCP 初建连接时，通讯双方要交换 MSS，以确定将要传输的数据的大小，MSS 值设置为外出接口上的 MTU 长度减去固定的 IP 首部和 TCP 首部长度。但是如果网络的中间设备的 MTU 要小于通信双方计算机的 MTU，那么数据在传输过程中就要分段。TCP 采用一个 Path MTU Discovery 机制来避免当不同网络的计算机要通信时出现的分段问题，即通信前先发送一个 IP 探测数据，数据包的大小为输出接口或者对端声明的 MSS 中的最小 MTU，这个数据包的 IP 报头“不分片”标志字段必须被设置为 1，以表明禁止数据包分段，如果 IP 探测器没有被分段到达了目的地，那么就会采用此探测器段中用到的 MTU 用作数据传输，如果发送端收到一个 ICMP 错误消息“不能分片”，一个表示数据包太大，那么发送端再发送一个数据包小一点的探测器，直到有个探测器顺利到达对端。

**2. 确认机制、序号机制、超时重传机制**

当 TCP 的连接建立好后，为保证数据传输的可靠，TCP 协议要求对传输的数据都进行确认，为保证确认的正常进行，TCP 协议首先对每一个分段都作了 32 位的编号，称为序列号。每一个分段都按照从起始号递增的顺序进行编号。TCP 协议通过序列号以及确认号来确保传输的可靠性，每一次传输数据时都会标明该段的编号，以便对方确认，同时在确认字段中对已收到的 TCP 分段确认。确认并不需要单独发包确认，可以放在传到对方的 TCP 分段中，在 TCP 协议中并不直接确认收到哪些分段，而是通知发送方下一次该发送哪一个分段，表示前面的分段都已收到；TCP 协议的确认通常采用延时几分之一秒后再做确认，而不是收到一个确认一个，接收端可能收到从 X 到 X+N 的 N 个后才开始确认，直接在确认字段中标 N+X+1，通知对方下一次直接传 N+X+1 分段，这样减少确认的次数以增加确认的效率。发送端发送一段数据后，会设置一个(RTO)重传超时计时器，当计时器时间内没有收到对方的确认消息，或者发送端收到 3 个重复 ACK 后就重发数据段。如果 M(M<N)分段在传输中出错，则确认 X+M 通知发送方从 X+M 开始重传 X+M 分段以及以后的所有分段。TCP 的确认和重传技术对每一个分段都有唯一的编号，这样当对方收到了重复的分段后容易区分，数据分段丢失后也容易定位重传的数据的编号。

**3. 缓冲传输机制**

当需要传输不大的数据时，发送端应用程序将一直等待，直到积累为一个大的数据块再进行传输。

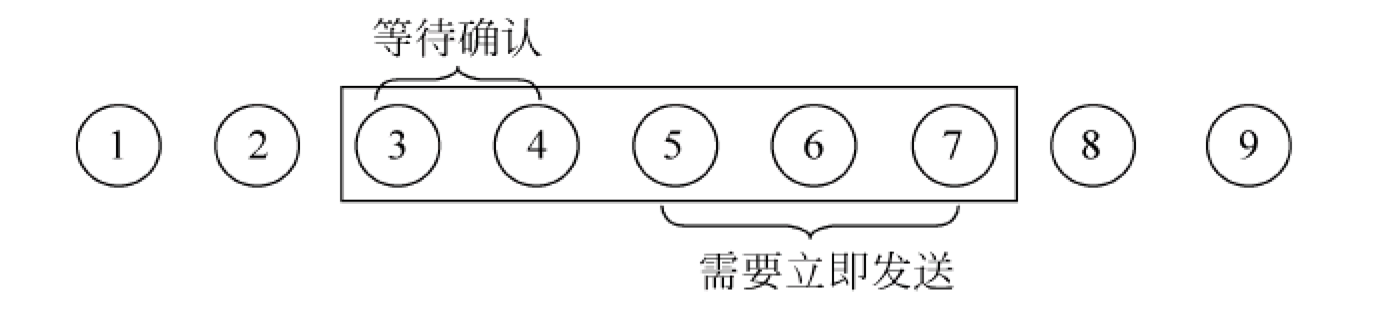
**4. 滑动窗口机制**

为了提高传输效率，TCP 在缓存上定义了一个窗口，发送端不必等收到确认再发送数据，发送数据的多少由这个窗口决定，这种机制叫滑动窗口机制。它将多个段放置到一个称为窗口的组中，并且所有的段都不需要等待确认消息，就可以作为一个整体发送出去，这个窗口叫做发送窗口，发送端的发送窗口值取接收 rwnd 和拥塞窗口 cwnd 中最小的一个。

在图 3-2 中，所有未被确认的段都位于窗口的左边，需要立即发送的段位于窗口的右边。当滑动窗口的左端到达最右端时就被称为零窗口，这标志着数据传输的结束。在图中可以看到，段 1 和段 2 是由接收端确认的段不在窗口中，段 3 和段 4 已经发送出去，但还在等待确认，在窗口的左边，段 5、段 6 和段 7 是窗口序列中即将要发送的下一段。当段 3 被确认释放后段 8 就进入窗口右端。这样当窗口中有被确认的段并释放后，窗口就会向前滑动。滑动窗口确保了发送计算机能够传输最大数据量的段。但是当接收端不能容纳大量数据段的话，就会发生拥塞。因此，接收端必须根据自己所能接收和缓冲的数据大小，控制发送端计算机的窗口大小，接收端将在确认段 TCP 头部的 Window 字段中指定窗口的大小。发送端收到这个信息后，可以调整自己的窗口尺寸。

**5. 延迟机制**

当发送端发送一个很大的段从而阻塞了接收端的缓冲区的时候，接收端就会发送一个



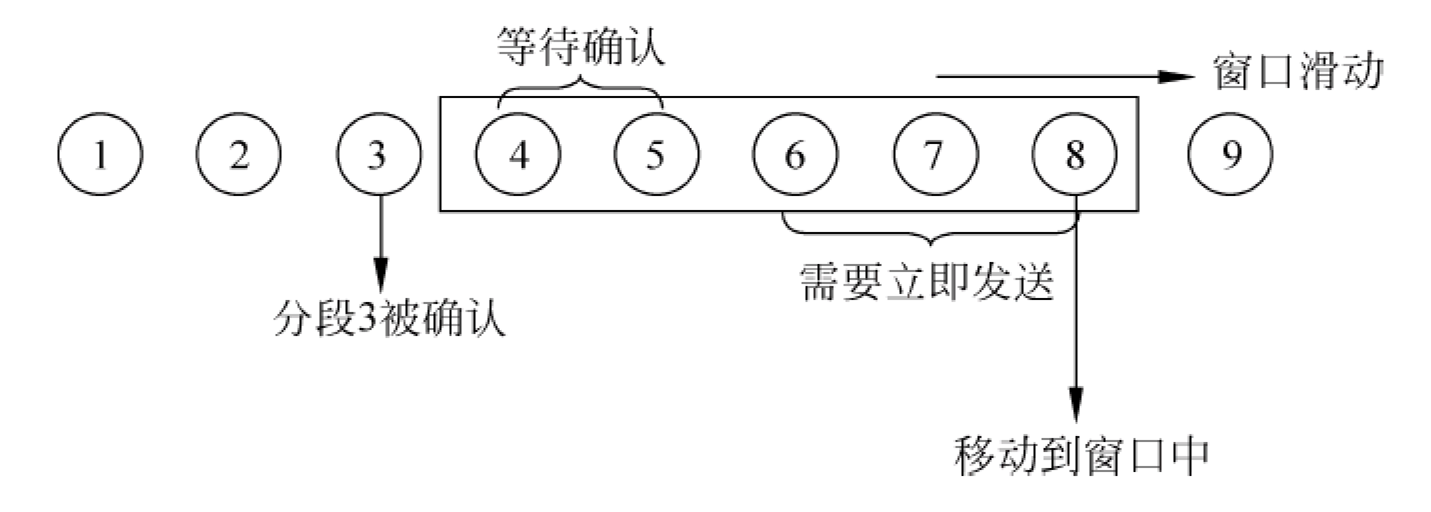




图 3-2

非常小的窗口应答，于是发送端就发送很小的数据段，这样造成网络资源利用率低，为了解决这个问题，TCP 用 Nagle 算法，限制发送端一收到确认就立即传输数据，发送端不传送小数据段，而是将要发送的段积聚成一个大段，这个大数据段的尺寸接近 MSS 值时，就一下子发送此段。这个方法通过积累大的数据段来缓解接收端的缓冲区阻塞，让接收方在接收数据前清空缓冲区，从而通告一个大的窗口。同样，接收端在收到数据段后也不立即发送确认，这个延迟，让接收端清空自己的一半缓冲区后，再发送确认消息，表明收到了所有的数据段。这样，更大的窗口就能被通告。

**注意：**建议延迟最大为 500ms。

#### 6. 估计往返时间

为了确定超时值，需要计算往返样本时间的加权平均值。

#### 7. 持续定时器

当缓冲区空间满时，接收端会发送一个窗口为 0 的确认。这个确认使得发送端停止继续传送数据，这时，TCP 就通过在发送端上创建一个持续定时器，在这个定时器到期后，发送一个新的窗口探测数据包，等待接收端返回带有一个窗口信息确认，如果这个确认中的窗口为 0，那么继续下个新的持续定时器，如果非 0，发送端就可以开始发送数据了。

#### 8. 保持活动定时器

由于 TCP 是面向连接的，因此，会出现连接建立后，没有数据传输这种情况。这时就需要保持活动定时器来检测这种连接，并释放连接。

#### 9. 拥塞窗口

拥塞窗口(cwnd)，拥塞控制的关键参数，它描述发送端在拥塞控制情况下一次最多能发送数据包的数量。发送端的发送窗口值不仅取决于接收端的可用缓存空间(rwnd，接收窗口)，还取决于网络的拥塞。为了处理与拥塞相关的问题，需要在发送系统中使用“拥塞窗

口”。当发现超时或收到 3 个相同 ACK 确认帧时，就发生了拥塞，这时，发送端通过减少自己的发送操作来做出响应，以便处理拥塞，发送端创建一个叫做拥塞窗口的窗口，段就基于拥塞窗口来发送，而不是基于接收端所通告的接收窗口。

**10. 慢启动和拥塞避免**

在现实的网络中，发送端和接收端之间存在多个路由器和速率较慢的链路，当一个快速链路向一个慢速链路发送数据时，会出现路由器来不及转发这些数据，这些数据占据了路由器的缓存，由于缓存资源有限，当缓存中的数据长度达到规定的最大长度时，所有到来的报文都被丢弃。对于 TCP 报文，由于大量的报文被丢弃，将造成 TCP 超时，从而引发 TCP 的慢启动和拥塞避免机制，使 TCP 减少报文的发送。

拥塞避免算法和慢启动算法需要对每个连接维持两个变量：一个拥塞窗口 cwnd 和一个慢启动门限 ssthresh。

慢启动算法在 cwnd＜ssthresh 时使用，拥塞避免算法在 cwnd＞ssthresh 时使用。当 cwnd 和 ssthresh 相等时，发送端既可以使用慢启动也可以使用拥塞避免。

慢启动算法初始设置 cwnd 为 1 个报文段即 cwnd＝1MSS，此后每收到一个确认，cwnd 就指数方式增长：发送 1 个报文段，然后是 2 个，接着是 4 个……

拥塞避免算法要求每次收到一个确认时将 cwnd 增加 1/cwnd。与慢启动的指数增加比起来，这是一种加性增长(additive increase)。

拥塞避免算法和慢启动算法是两个目的不同、独立的算法。在实际中这两个算法通常在一起实现。

慢启动和拥塞避免工作流程如下：

(1) 对一个给定的连接，初始化 cwnd 为 1 个报文段，ssthresh 为 65 535 个字节。

(2) TCP 发送端发送的数据不能超过 cwnd 和接收方通告窗口的大小。cwnd 是发送方使用的流量控制，而通告窗口则是接收方进行的流量控制。前者是发送方感受到的网络拥塞的估计，而后者则与接收方在该连接上的可用缓存大小有关。

(3) 当拥塞发生时(超时或收到重复确认)，ssthresh 被设置为当前窗口大小的一半(cwnd 和接收方通告窗口大小的最小值，但最少为 2 个报文段)。如果是超时引起了拥塞，则 cwnd 被设置为 1 个报文段(这就是慢启动)。

(4) 当新的数据被对方确认时，就增加 cwnd，但增加的方法依赖于是否正在进行慢启动或拥塞避免。如果 cwnd 小于或等于 ssthresh，则正在进行慢启动，否则正在进行拥塞避免。慢启动一直持续到 cwnd 等于 ssthresh 时停止，然后转为执行拥塞避免。慢启动时，cwnd 呈指数增加，拥塞避免时，cwnd 呈线性增加也叫加法增加。

举个以上工作流程的例子。假设当 cwnd 为 32 个报文段时就会发生拥塞。于是设置 ssthresh 为 16 个报文段，而 cwnd 为 1 个报文段。在时刻 0 发送了一个报文段，并假定在时刻 1 接收到它的 ACK，此时 cwnd 增加为 2。接着发送了 2 个报文段，并假定在时刻 2 接收到它们的 ACK，于是 cwnd 增加为 4，就这样，接收到下一个 ACK 时，cwnd 就增加为 8，就这样每收到一个 ACK，cwnd 就指数增加。这种指数增加算法一直进行到 cwnd 等于 ssthresh 时才停止，从该时刻起进入拥塞避免，cwnd 以线性方式增加，这种方式是在一个往返时间内(不管在这个 RTT 中收到了多少个 ACK)，cwnd 最多增加 1 个报文段。

**11．快速重传**

如果收到一连串 3 个或 3 个以上的重复 ACK，就非常可能是一个报文段丢失了。这时就重传丢失的数据包文段，而无需等待超时定时器溢出。这就是快速重传。

## 3.1.3 TCP 数据包分析

介绍完了 TCP 的原理，下面来看一个 FTP 应用的 TCP 包的实例。

**1．Source port**

它指的是应用程序所用的 16 位 TCP 端口号

如图 3-3 所示，一般 FTP 会开放 Port 21 用来接受控制命令，当传送数据的时候，FTP Server 会自动打开 Port 20 通过该端口传送。从图中看到源端口为 20。

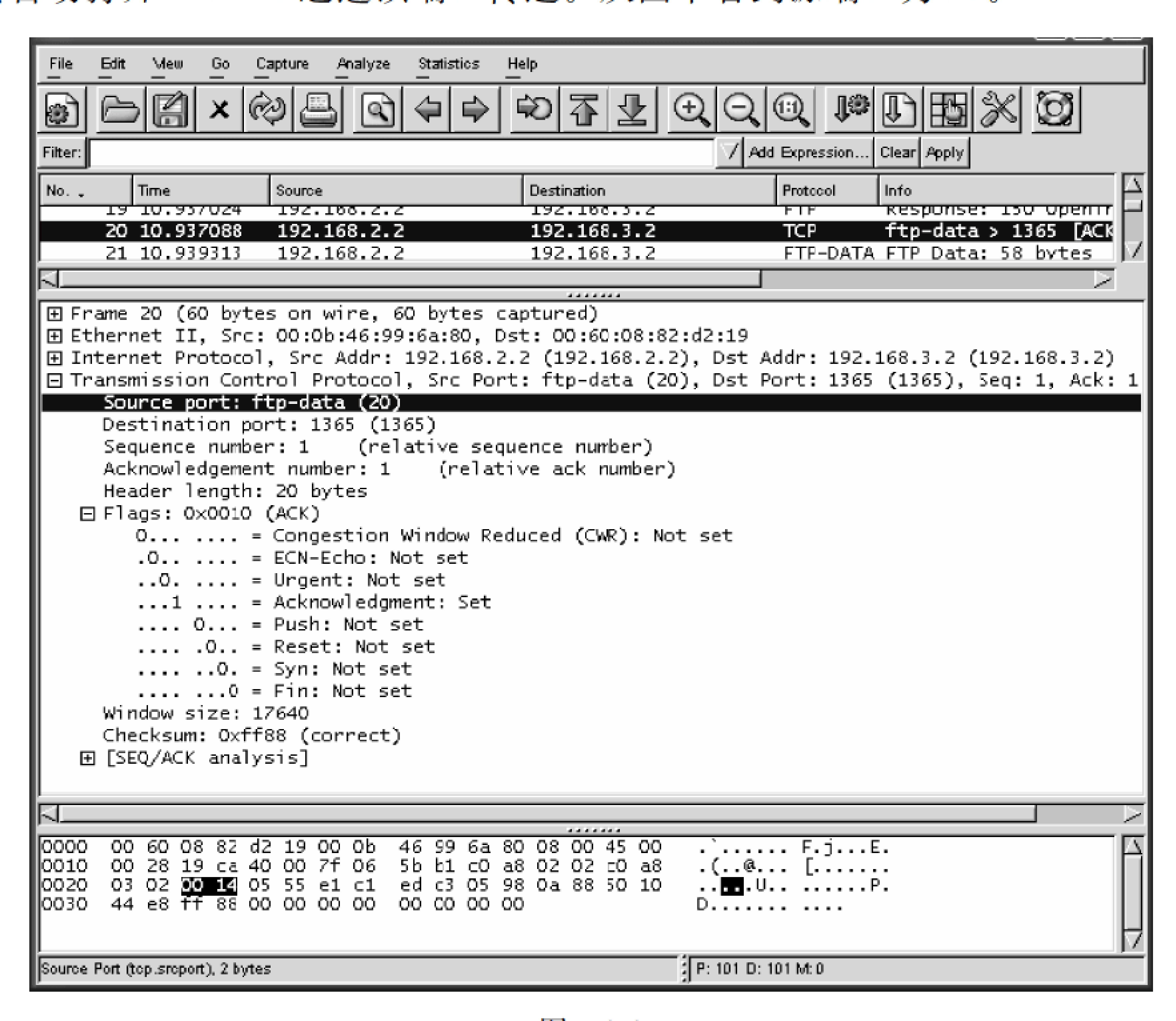



图 3-3

**2．Destination port**

它指的是目的系统上的应用程序所用的 16 位 TCP 端口号。

如图 3-4 所示，客户端使用端口 1365 用来接受数据。

**3．Sequence number**

如图 3-5 所示，通常指定分配到当前信息中的数据首字节的序号。在连接建立阶段，该字段用于设置传输中的初始序列号。

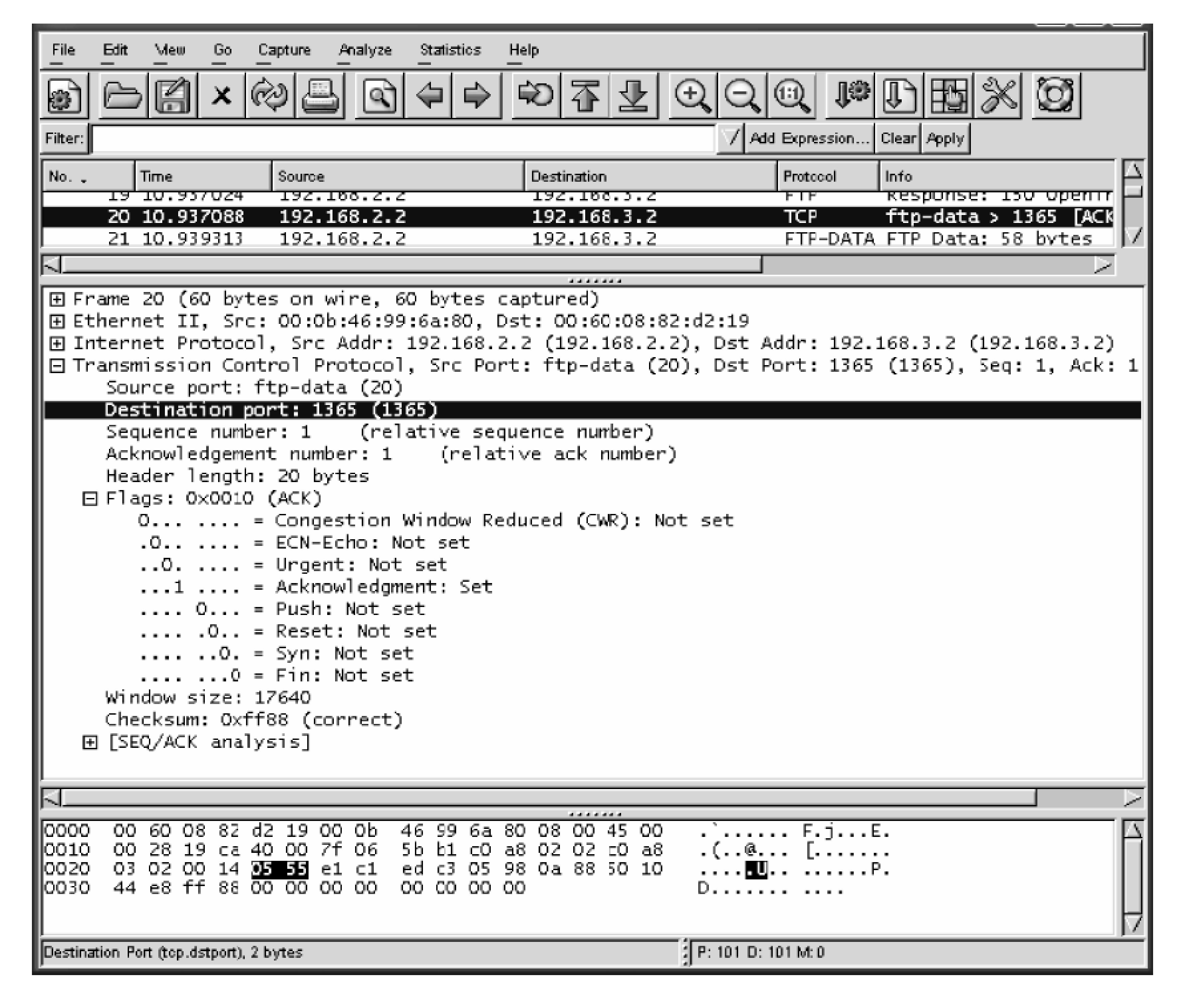

File Edit View Go Capture Analyze Statistics Help
Filter:
Add Expression... Clear Apply
No. Time Source Destination Protocol Info
20 10.937088 192.168.2.2 192.168.3.2 TCP ftp-data > 1365 [ACK
21 10.939313 192.168.2.2 192.168.3.2 FTP-DATA FTP Data: 58 bytes
Frame 20 (60 bytes on wire, 60 bytes captured)
Ethernet II, Src: 00:0b:46:99:6a:80, Dst: 00:60:08:82:d2:19
Internet Protocol, Src Addr: 192.168.2.2 (192.168.2.2), Dst Addr: 192.168.3.2 (192.168.3.2)
Transmission Control Protocol, Src Port: ftp-data (20), Dst Port: 1365 (1365), Seq: 1, Ack: 1
Source port: ftp-data (20)
Destination port: 1365 (1365)
Sequence number: 1 (relative sequence number)
Acknowledgement number: 1 (relative ack number)
Header length: 20 bytes
Flags: 0x0010 (ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 0... = Push: Not set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 17640
Checksum: 0xff88 (correct)
[SEQ/ACK analysis]
0000 00 60 08 82 d2 19 00 0b 46 99 6a 80 08 00 45 00
0010 00 28 19 ca 40 00 7f 06 5b b1 c0 a8 02 02 c0 a8
0020 03 02 00 14 05 55 e1 c1 ed c3 05 98 0a 88 50 10
0030 44 e8 ff 88 00 00 00 00 00 00 00 00
Destination Port (tcp.dstport), 2 bytes
P: 101 D: 101 M: 0

图　3-4

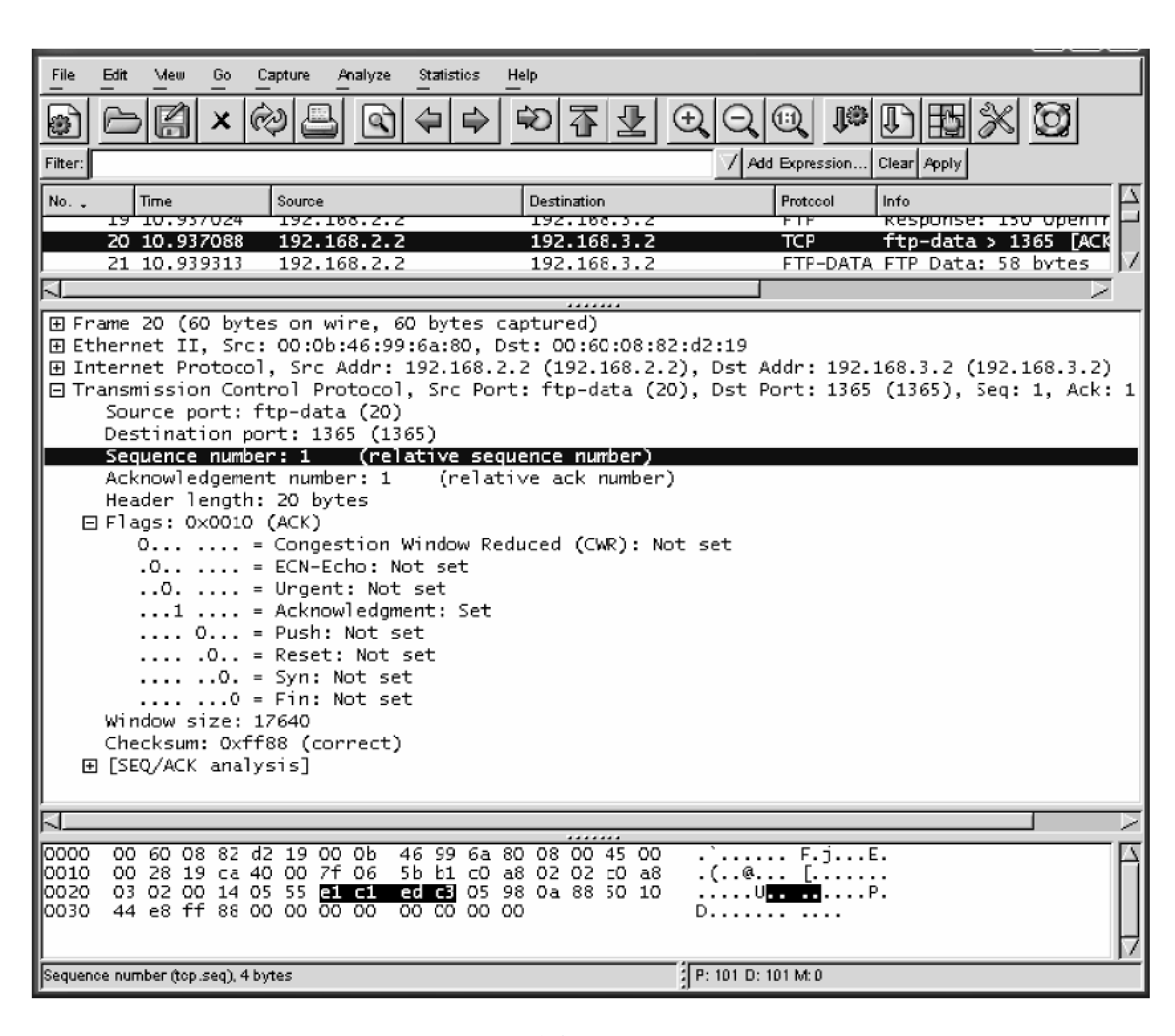

File Edit View Go Capture Analyze Statistics Help
Filter:
Add Expression... Clear Apply
No. Time Source Destination Protocol Info
20 10.937088 192.168.2.2 192.168.3.2 TCP ftp-data > 1365 [ACK
21 10.939313 192.168.2.2 192.168.3.2 FTP-DATA FTP Data: 58 bytes
Frame 20 (60 bytes on wire, 60 bytes captured)
Ethernet II, Src: 00:0b:46:99:6a:80, Dst: 00:60:08:82:d2:19
Internet Protocol, Src Addr: 192.168.2.2 (192.168.2.2), Dst Addr: 192.168.3.2 (192.168.3.2)
Transmission Control Protocol, Src Port: ftp-data (20), Dst Port: 1365 (1365), Seq: 1, Ack: 1
Source port: ftp-data (20)
Destination port: 1365 (1365)
Sequence number: 1 (relative sequence number)
Acknowledgement number: 1 (relative ack number)
Header length: 20 bytes
Flags: 0x0010 (ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 0... = Push: Not set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 17640
Checksum: 0xff88 (correct)
[SEQ/ACK analysis]
0000 00 60 08 82 d2 19 00 0b 46 99 6a 80 08 00 45 00
0010 00 28 19 ca 40 00 7f 06 5b b1 c0 a8 02 02 c0 a8
0020 03 02 00 14 05 55 e1 c1 ed c3 05 98 0a 88 50 10
0030 44 e8 ff 88 00 00 00 00 00 00 00 00
Sequence number (tcp.seq), 4 bytes
P: 101 D: 101 M: 0

图　3-5

TCP 发送的每个字节数据的给定序列号都不相同。当数据存储在片段中等待发送时，该片段中数据的第一个字节的序列号就放入 Sequence number 字段。e1c1edc3 用十进制表示为 3 787 582 915，二进制为 1110 0001 1100 0001 1110 1101 1100 0011。

如果将字节流看作在两个应用程序间的单向流动，则 TCP 用序号对每个字节进行计数。序号是 32 位的无符号数，序号到达 $2^{32}-1$ 后又从 0 开始。

### 4. Acknowledgment number

在握手阶段，确认序号将发送方的序列号加 1 作为回答，在数据传输阶段，确认序号将发送方的序列号加发送的数据大小作为回答，表示确实收到这些数据，如图 3-6 所示。

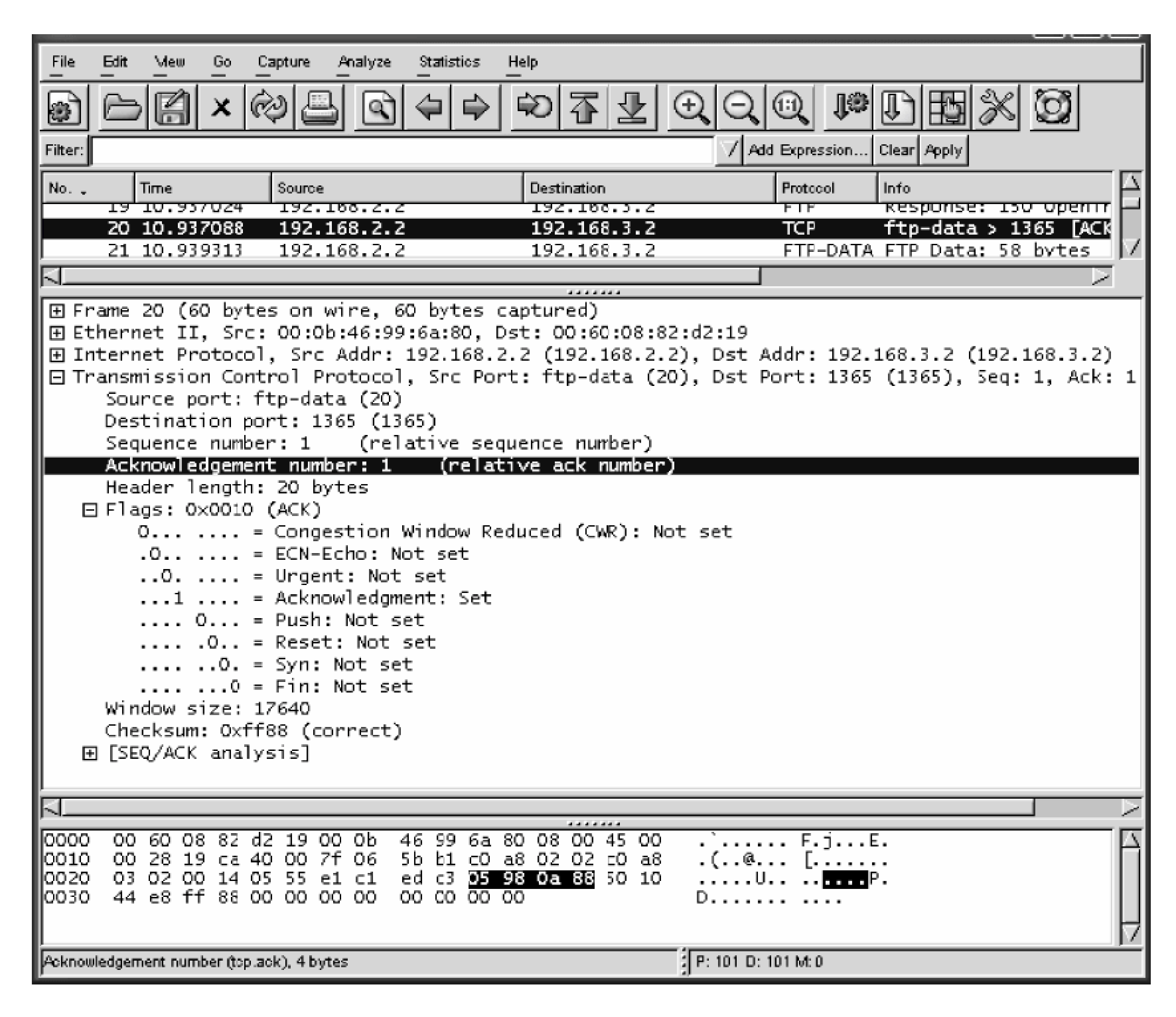



图 3-6

### 5. Header length

表示 20 个字节，计算方法和作用与 IP 报头相同，如图 3-7 所示。

### 6. Flags

如图 3-8 所示，这个字段包含以下几项。

- URG：紧急指针是否有效。用到的时候值为 1，表示紧急指针有效。
- ACK：为 1 时表示确认号(Acknowledgment number)为合法，为 0 时表示数据段不包含确认信息，确认号被忽略。

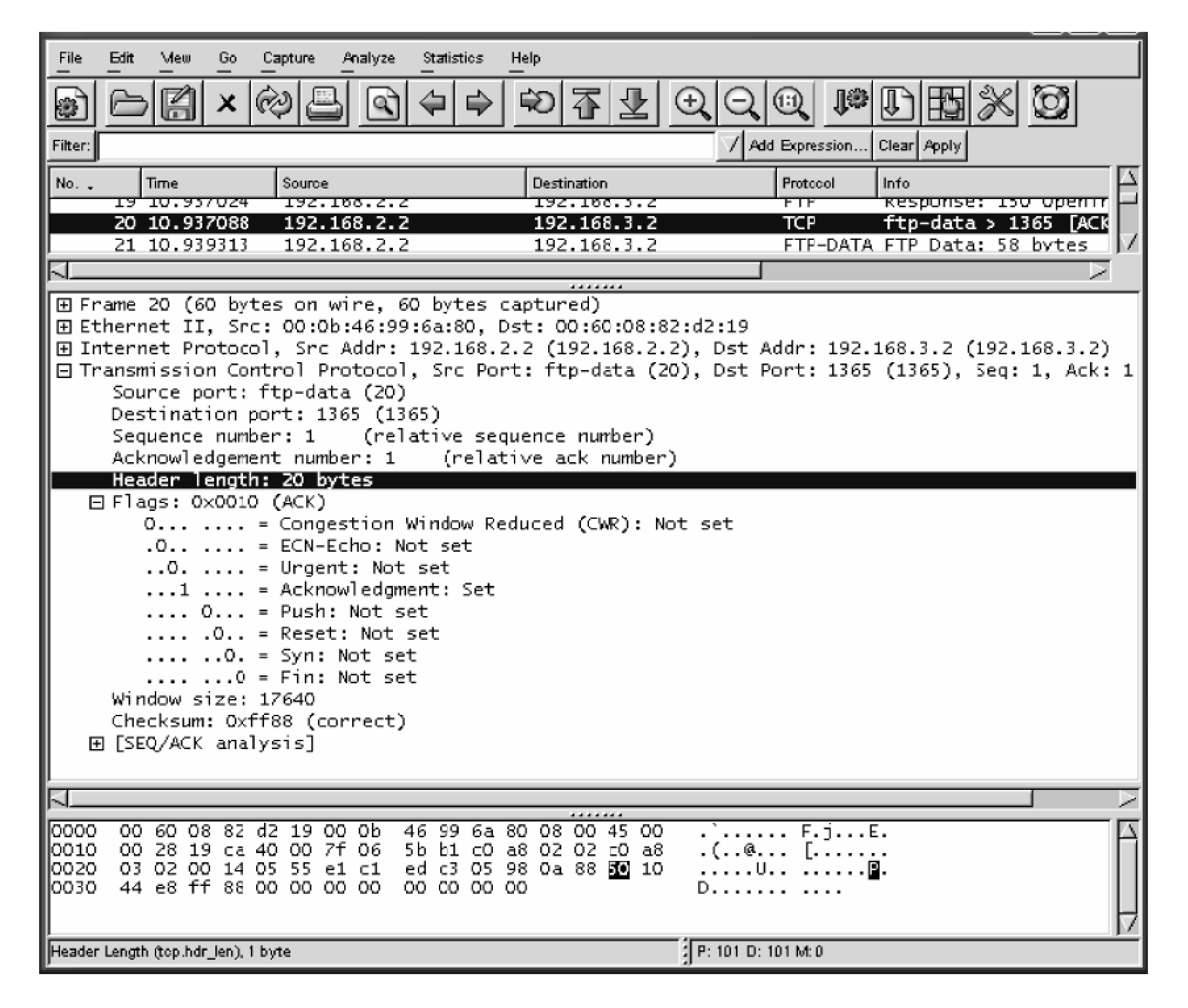

File Edit View Go Capture Analyze Statistics Help
Filter:
Add Expression... Clear Apply
No. Time Source Destination Protocol Info
20 10.937088 192.168.2.2 192.168.3.2 TCP ftp-data > 1365 [ACK
21 10.939313 192.168.2.2 192.168.3.2 FTP-DATA FTP Data: 58 bytes
Frame 20 (60 bytes on wire, 60 bytes captured)
Ethernet II, Src: 00:0b:46:99:6a:80, Dst: 00:60:08:82:d2:19
Internet Protocol, Src Addr: 192.168.2.2 (192.168.2.2), Dst Addr: 192.168.3.2 (192.168.3.2)
Transmission Control Protocol, Src Port: ftp-data (20), Dst Port: 1365 (1365), Seq: 1, Ack: 1
Source port: ftp-data (20)
Destination port: 1365 (1365)
Sequence number: 1 (relative sequence number)
Acknowledgement number: 1 (relative ack number)
Header length: 20 bytes
Flags: 0x0010 (ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 0... = Push: Not set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 17640
Checksum: 0xff88 (correct)
[SEQ/ACK analysis]
0000 00 60 08 82 d2 19 00 0b 46 99 6a 80 08 00 45 00
0010 00 28 19 ca 40 00 7f 06 5b b1 c0 a8 02 02 c0 a8
0020 03 02 00 14 05 55 e1 c1 ed c3 05 98 0a 88 50 10
0030 44 e8 ff 88 00 00 00 00 00 00 00 00
Header Length (tcp.hdr_len), 1 byte
P: 101 D: 101 M: 0

图　3-7

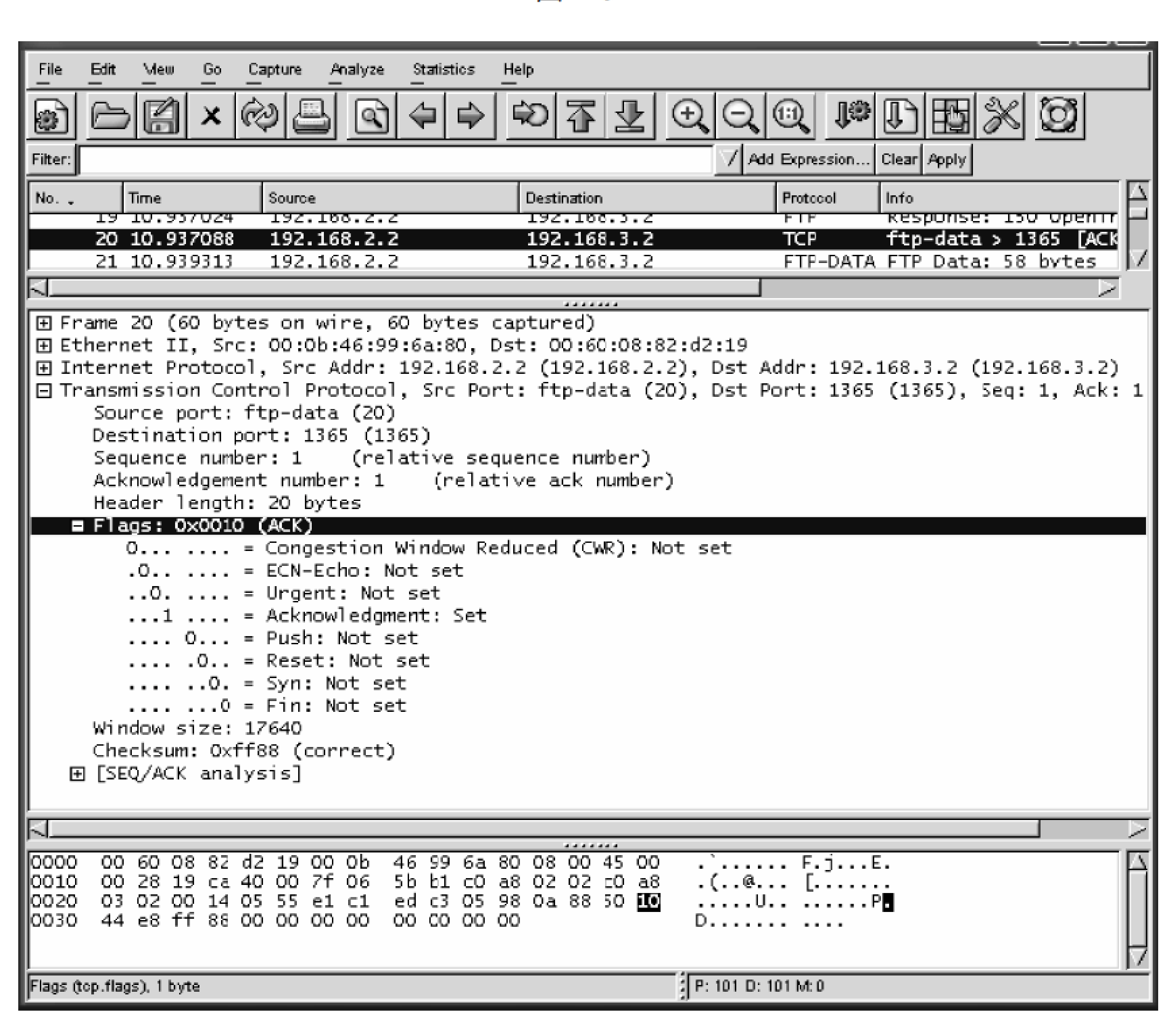

File Edit View Go Capture Analyze Statistics Help
Filter:
Add Expression... Clear Apply
No. Time Source Destination Protocol Info
20 10.937088 192.168.2.2 192.168.3.2 TCP ftp-data > 1365 [ACK
21 10.939313 192.168.2.2 192.168.3.2 FTP-DATA FTP Data: 58 bytes
Frame 20 (60 bytes on wire, 60 bytes captured)
Ethernet II, Src: 00:0b:46:99:6a:80, Dst: 00:60:08:82:d2:19
Internet Protocol, Src Addr: 192.168.2.2 (192.168.2.2), Dst Addr: 192.168.3.2 (192.168.3.2)
Transmission Control Protocol, Src Port: ftp-data (20), Dst Port: 1365 (1365), Seq: 1, Ack: 1
Source port: ftp-data (20)
Destination port: 1365 (1365)
Sequence number: 1 (relative sequence number)
Acknowledgement number: 1 (relative ack number)
Header length: 20 bytes
Flags: 0x0010 (ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 0... = Push: Not set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 17640
Checksum: 0xff88 (correct)
[SEQ/ACK analysis]
0000 00 60 08 82 d2 19 00 0b 46 99 6a 80 08 00 45 00
0010 00 28 19 ca 40 00 7f 06 5b b1 c0 a8 02 02 c0 a8
0020 03 02 00 14 05 55 e1 c1 ed c3 05 98 0a 88 50 10
0030 44 e8 ff 88 00 00 00 00 00 00 00 00
Flags (tcp.flags), 1 byte
P: 101 D: 101 M: 0

图　3-8

- Push，PUSH 标志的数据，为 1 时请求的数据段在接收方得到后就可直接送到应用程序，而不必等到缓冲区满时才传送。
- RST：为 1 的时候，表示请求重新连接。用于复位因某种原因引起出现的错误连接，也用来拒绝非法数据和请求。如果接收到 RST 置位时，通常发生了某些错误。
- SYN：为 1 的时候，表示请求建立连接。当建立一个新的连接时，SYN 位变 1。在连接请求中，SYN＝1，ACK＝0；连接响应时，SYN＝1，ACK＝1。即用 SYN 和 ACK 来区分 Connection Request 和 Connection Accepted。
- FIN：为 1 的时候，表示释放连接，表明发送方已经没有数据发送了。

**注意**：在图 3-8 的 Flags 字段中还看到了另外两位，一个是 cwr(congestion window reduced)，另一个是 ENC-Echo，这两个标志是在 RFC 3168 中定义的，它定义了 Flags 字段为 8 位，为了增强对 ECN(Explicit Congestion Notification)的支持。而 RFC 793 中则定义了 Flags 字段为 6 位，就是上面的 URG、ACK、PSH、RST、SYN、FIN。

### 7. Window size

说明发送系统上的可用的接收缓冲区空间的数量，以字节表示，如图 3-9 所示。

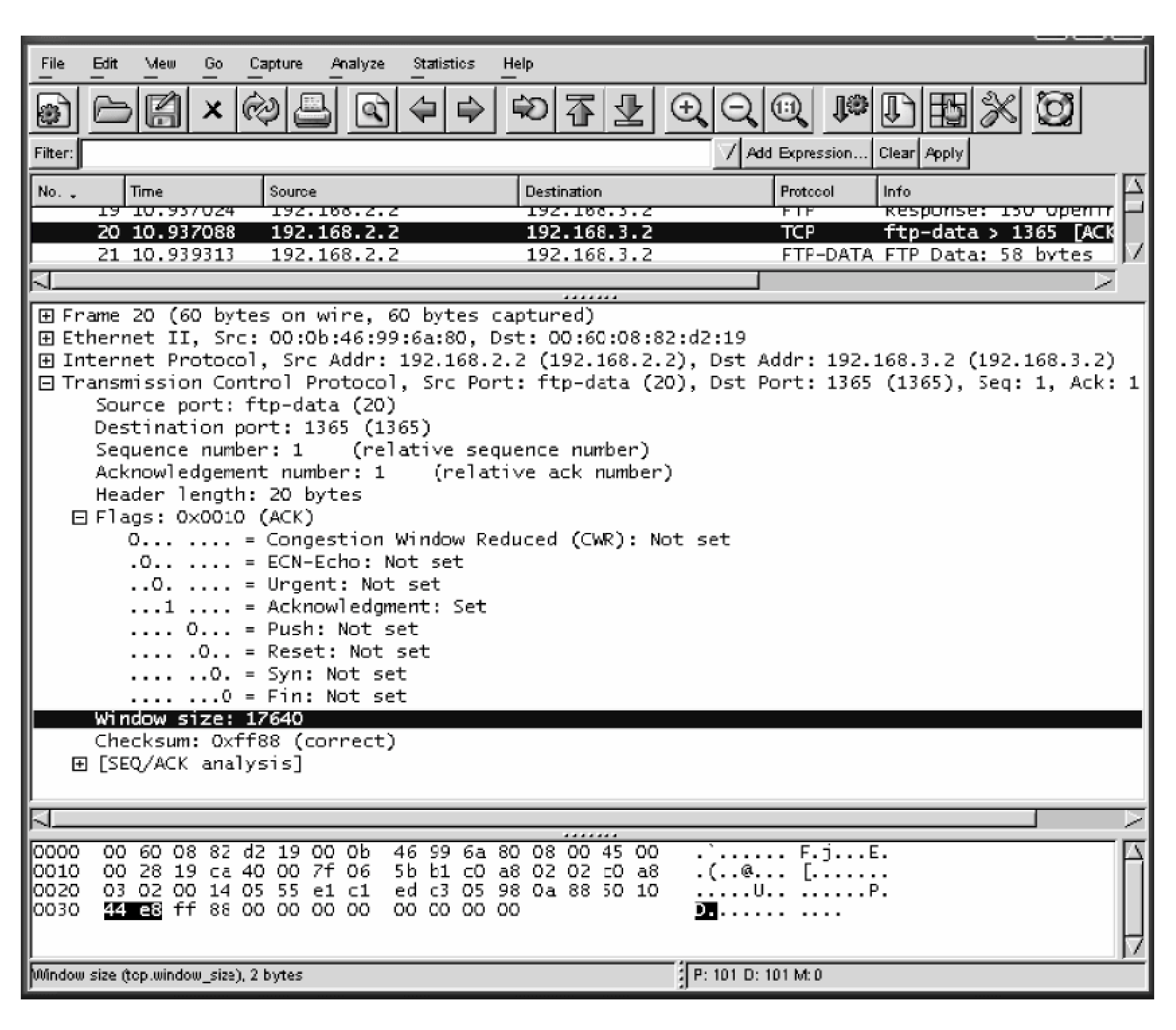



图 3-9

TCP 流控制的一个主要部分就是让每个系统跟踪其他系统的接收缓冲区空间。通告一个小的接收缓冲区，虚电路的终端能够有效地强迫另一终端暂时停止发送数据，而增加缓冲区大小使得另一终端能够发送较多数据而不需要等待确认。接收缓冲区的大小存储在每个 TCP 片段的首部的 window 片段中，这样可以连续地交换缓冲区状态信息。

**8. Checksum**

用于存储 TCP 片段的校验和,包括首部和主体部分。校验和允许目的系统可以验证 TCP 片段的内容并能够测试可能的破坏,如图 3-10 所示。

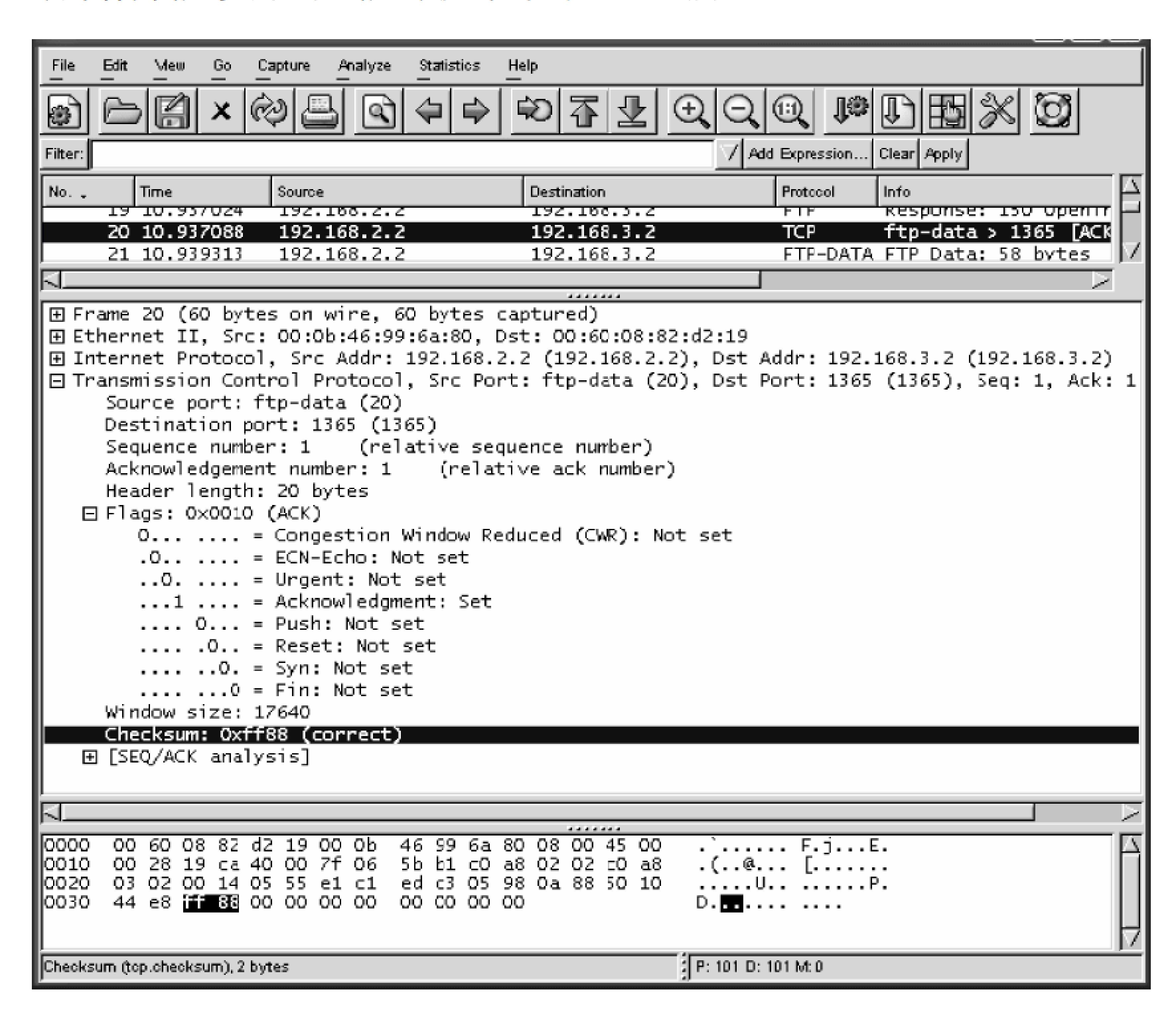



图　3-10

## 3.1.4　TCP 三次“握手”

TCP 是一个面向连接的协议,无论哪一方向另一方发送数据之前,都必须先在双方之间建立一条连接。

通常一条 TCP 连接的建立,需要经过常说的三次“握手”。

(1) 请求端(通常称为客户)发送一个 SYN 报文段指明客户打算连接的服务器的端口,以及初始序号 ISN,这个 SYN 段称为报文 1。

(2) 服务器发回包含服务器的初始序号的 SYN 报文段(报文段 2)作为应答。同时,将确认序号设置为客户的 ISN 加 1 以对客户的 SYN 报文段进行确认。一个 SYN 将占用一个序号。

(3) 客户必须将确认序号设置为服务器的 ISN 加 1 以对服务器的 SYN 报文进行确认(报文段 3)。

连接建成后,这个连接将一直保持活动状态,直到超时或者任何一方发出一个 FIN(结束)信号下面通过实例来看看三次“握手”的过程。

第一次握手：可以看到 192.168.3.2 用端口 1364 向 FTP 服务器 192.168.2.2 的 21 端口发送一个连接请求。这个报文段的序号为 0,如图 3-11 所示。

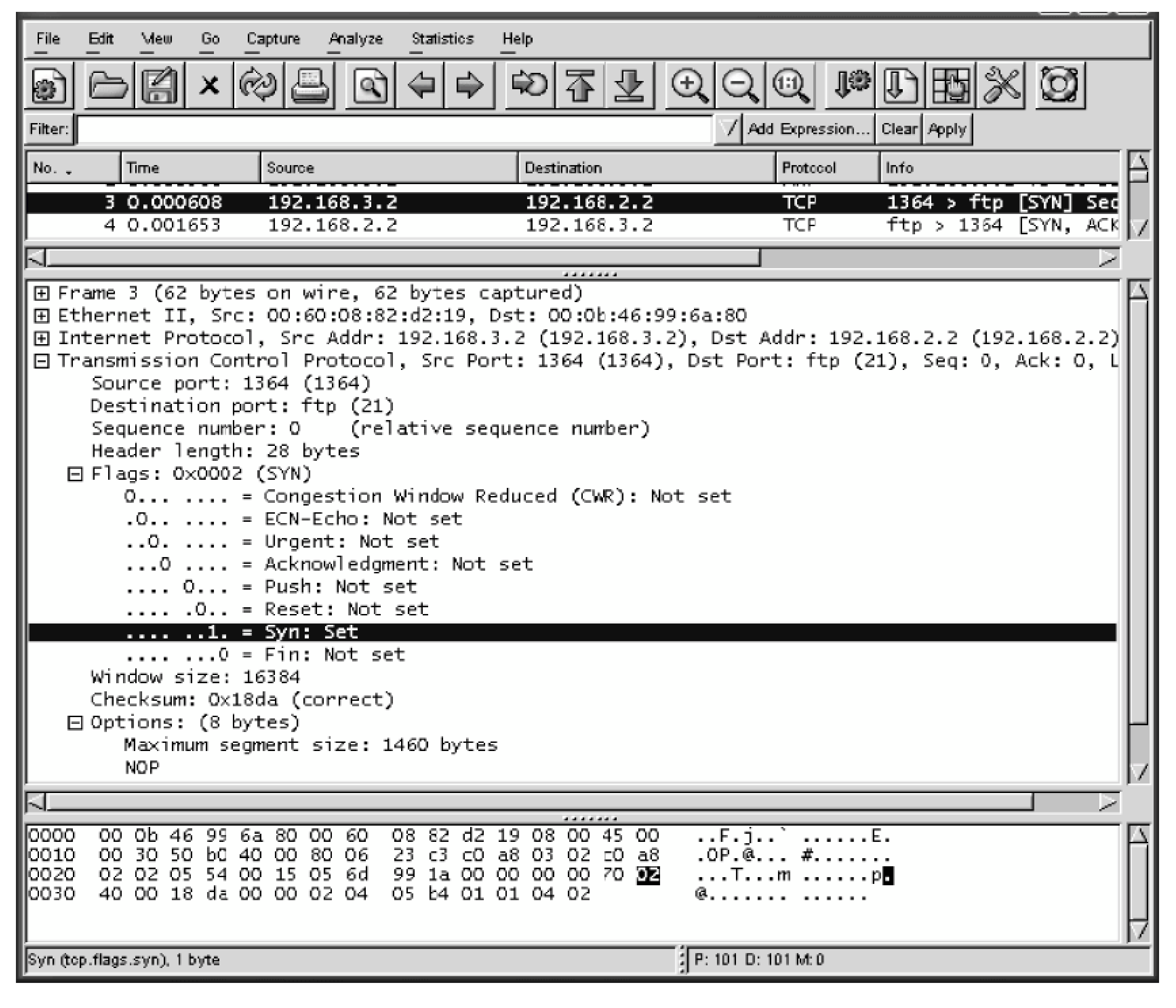



图 3-11

**注意**：MSS 字段为最大报文长度表示 TCP 传往另一端的最大块数据的长度。当一个连接建立时，连接的双方都要通告各自的 MSS。MSS 默认值为 536，如果没有分片发生，MSS 还是越大越好，MSS 值设置为外出接口上的 MTU 长度减去固定的 IP 首部和 TCP 首部长度，比如，一个以太网的 MSS 值可达 1460 字节。MSS 字段仅出现在连接建立时。

第二次握手：FTP 服务器用 21 端口向客户端 192.168.3.2 的端口 1364 确认刚才的连接请求。这个报文段的序号为 0，确认序号为图 3-11 中客户端发送的报文段序号＋1，也就是 0＋1＝1，如图 3-12 所示。

第三次握手：客户端发送一个带序号的报文对服务刚才发送的报文进行确认。这次发送的报文的序列号为 1，确认序号为图 3-12 中服务器发送的报文段序号＋1，也就是 0＋1，如图 3-13 所示。

### 3.1.5 TCP 连接的终止

终止一个连接需要经过 4 次“握手”。这由 TCP 的半关闭(half-close)造成的。由于一个 TCP 连接是全双工的(即数据在两个方向上能同时传递)，因此每个方向必须单独地进行关闭，这个原则就是当一方完成它的数据发送任务后就能发送一个 FIN 来终止这个方向连接。当一端收到一个 FIN，它必须通知应用层另一端已经终止了那个方向的数据传送。发送 FIN 通常是应用层进行关闭的结果。

收到一个 FIN 只意味着在这一方向上没有数据流动。由于 TCP 连接是全双工的，一个 TCP 连接在收到一个 FIN 后仍能发送数据。而这对利用半关闭的应用来说是可能的，尽管在实际应用中只有很少的 TCP 应用程序这样做。TCP 连接的终止过程如下。

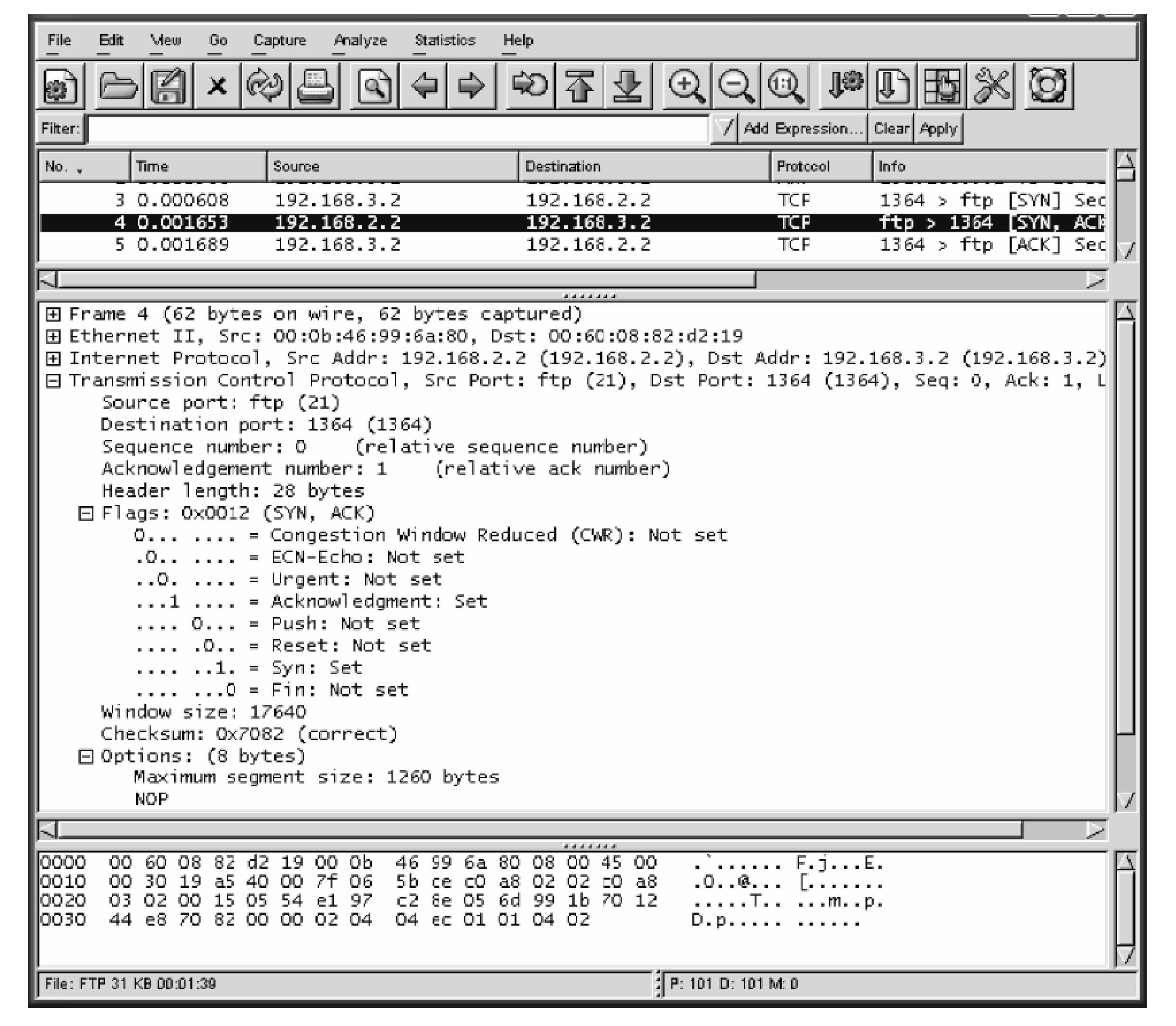

File Edit View Go Capture Analyze Statistics Help
Filter: Add Expression... Clear Apply
No. Time Source Destination Protocol Info
3 0.000608 192.168.3.2 192.168.2.2 TCP 1364 > ftp [SYN] Sec
4 0.001653 192.168.2.2 192.168.3.2 TCP ftp > 1364 [SYN, ACK
5 0.001689 192.168.3.2 192.168.2.2 TCP 1364 > ftp [ACK] Sec
Frame 4 (62 bytes on wire, 62 bytes captured)
Ethernet II, Src: 00:0b:46:99:6a:80, Dst: 00:60:08:82:d2:19
Internet Protocol, Src Addr: 192.168.2.2 (192.168.2.2), Dst Addr: 192.168.3.2 (192.168.3.2)
Transmission Control Protocol, Src Port: ftp (21), Dst Port: 1364 (1364), Seq: 0, Ack: 1, L
Source port: ftp (21)
Destination port: 1364 (1364)
Sequence number: 0 (relative sequence number)
Acknowledgement number: 1 (relative ack number)
Header length: 28 bytes
Flags: 0x0012 (SYN, ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 0... = Push: Not set
.... .0.. = Reset: Not set
.... ..1. = Syn: Set
.... ...0 = Fin: Not set
Window size: 17640
Checksum: 0x7082 (correct)
Options: (8 bytes)
Maximum segment size: 1260 bytes
NOP
0000 00 60 08 82 d2 19 00 0b 46 99 6a 80 08 00 45 00 .`...... F.j...E.
0010 00 30 19 a5 40 00 7f 06 5b ce c0 a8 02 02 c0 a8 .0..@... [.......
0020 03 02 00 15 05 54 e1 97 c2 8e 05 6d 99 1b 70 12 .....T.. ...m..p.
0030 44 e8 70 82 00 00 02 04 04 ec 01 01 04 02 D.p..... ......
File: FTP 31 KB 00:01:39
P: 101 D: 101 M: 0

图 3-12

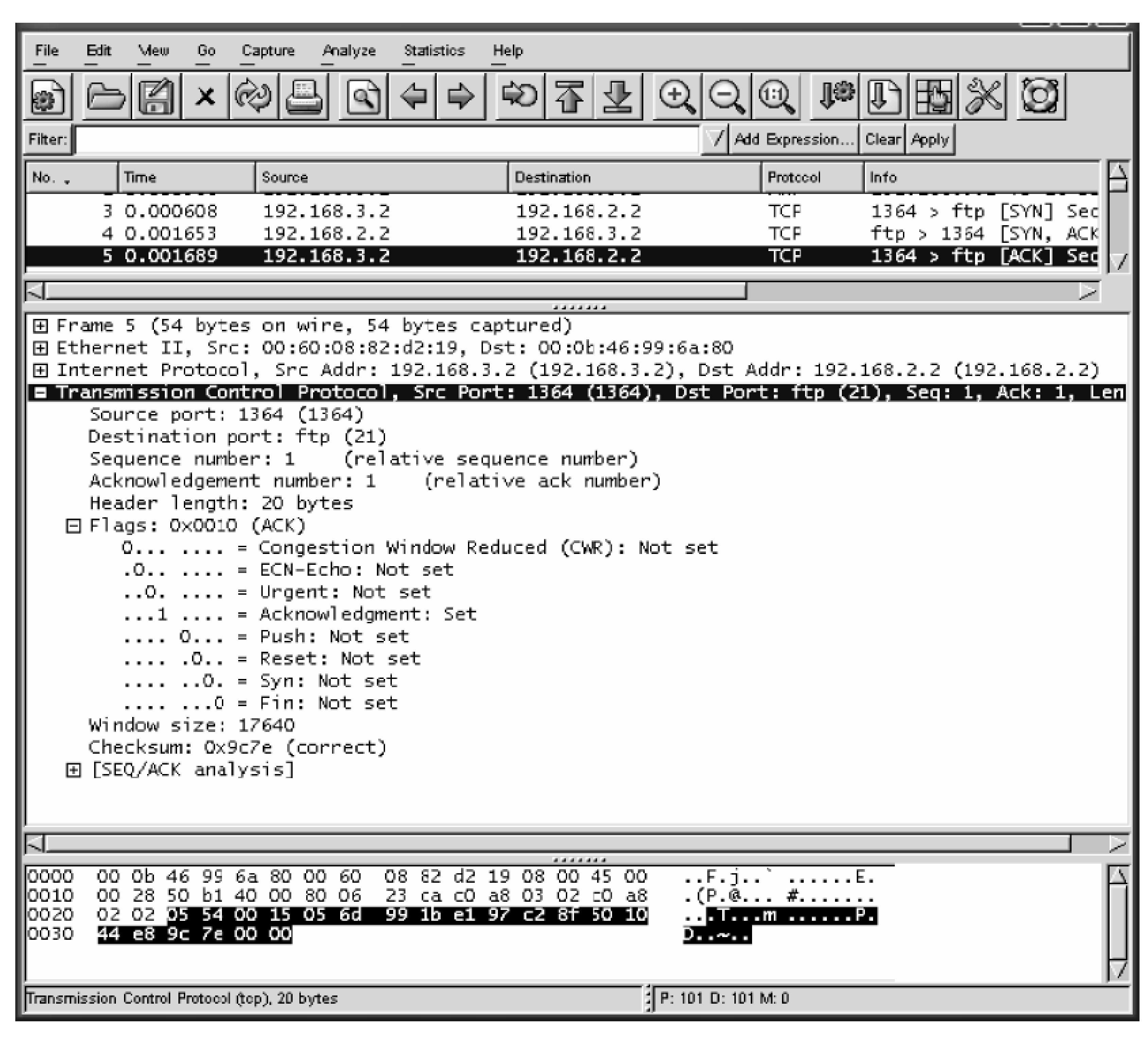

File Edit View Go Capture Analyze Statistics Help
Filter: Add Expression... Clear Apply
No. Time Source Destination Protocol Info
3 0.000608 192.168.3.2 192.168.2.2 TCP 1364 > ftp [SYN] Sec
4 0.001653 192.168.2.2 192.168.3.2 TCP ftp > 1364 [SYN, ACK
5 0.001689 192.168.3.2 192.168.2.2 TCP 1364 > ftp [ACK] Sec
Frame 5 (54 bytes on wire, 54 bytes captured)
Ethernet II, Src: 00:60:08:82:d2:19, Dst: 00:0b:46:99:6a:80
Internet Protocol, Src Addr: 192.168.3.2 (192.168.3.2), Dst Addr: 192.168.2.2 (192.168.2.2)
Transmission Control Protocol, Src Port: 1364 (1364), Dst Port: ftp (21), Seq: 1, Ack: 1, Len
Source port: 1364 (1364)
Destination port: ftp (21)
Sequence number: 1 (relative sequence number)
Acknowledgement number: 1 (relative ack number)
Header length: 20 bytes
Flags: 0x0010 (ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 0... = Push: Not set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 17640
Checksum: 0x9c7e (correct)
[SEQ/ACK analysis]
0000 00 0b 46 99 6a 80 00 60 08 82 d2 19 08 00 45 00 ..F.j..` ......E.
0010 00 28 50 b1 40 00 80 06 23 ca c0 a8 03 02 c0 a8 .(P.@... #.......
0020 02 02 05 54 00 15 05 6d 99 1b e1 97 c2 8f 50 10 ...T...m ......P.
0030 44 e8 9c 7e 00 00 D..~..
Transmission Control Protocol (tcp), 20 bytes
P: 101 D: 101 M: 0

图 3-13

（1）首先进行关闭的一方(即发送第一 FIN，一般是客户端)将执行主动关闭，而另一方(收到这个 FIN)执行被动关闭。通常一方完成主动关闭而另一方完成被动关闭。比如输入 bye 命令后即导致 TCP 客户端发送一个 FIN，用来关闭从客户端到服务器的数据传送。

（2）当接受方(服务器端)收到关闭方发送的 FIN，TCP 服务器向应用程序传送一个文件结束符，然后它发回一个 ACK，确认序号为收到的序号加 1，和 SYN 一样，一个 FIN 将占用一个序号。

（3）服务器程序就关闭它的连接，它的 TCP 端发送另一个 FIN。

（4）当客户端收到服务端发送的 FIN，客户就必须发回一个确认，并将确认序号设置为收到序号加 1。

下面通过实例来看看终止一个 TCP 连接所要经过的四次“握手”。

在客户端上输入 quit 来终止 FTP，如图 3-14 所示。

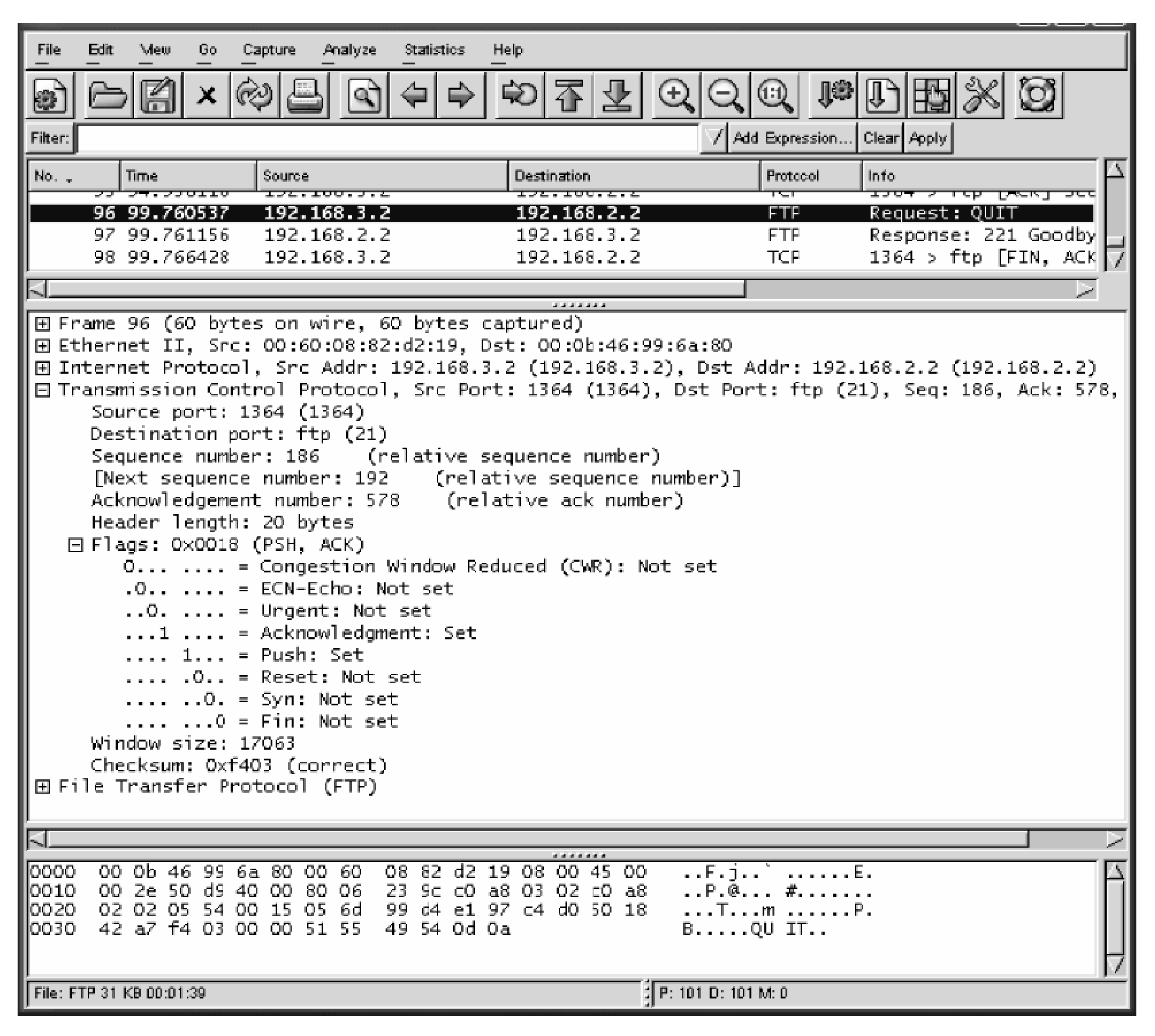



图 3-14

第一次握手：客户端用端口 1364 对 FTP 服务器端口 21 发送一个序号为 192 的 FIN 报文，如图 3-15 所示。

第二次握手：FTP 服务器端用端口 21 对客户端的端口 1364 发送一个序号为 592 的确认报文，它的 ACK 序号为 193(192＋1)，如图 3-16 所示。

第三次握手：服务器端又发送了一个序号为 592 的 FIN 报文，可以看到这个报文的序号和 ACK 序号和上面一个报文一样，如图 3-17 所示。

第四次握手：客户端的端口 21 发送一个序号为 193 的确认报文，它的 ACK 序号为 593(592＋1)，如图 3-18 所示。

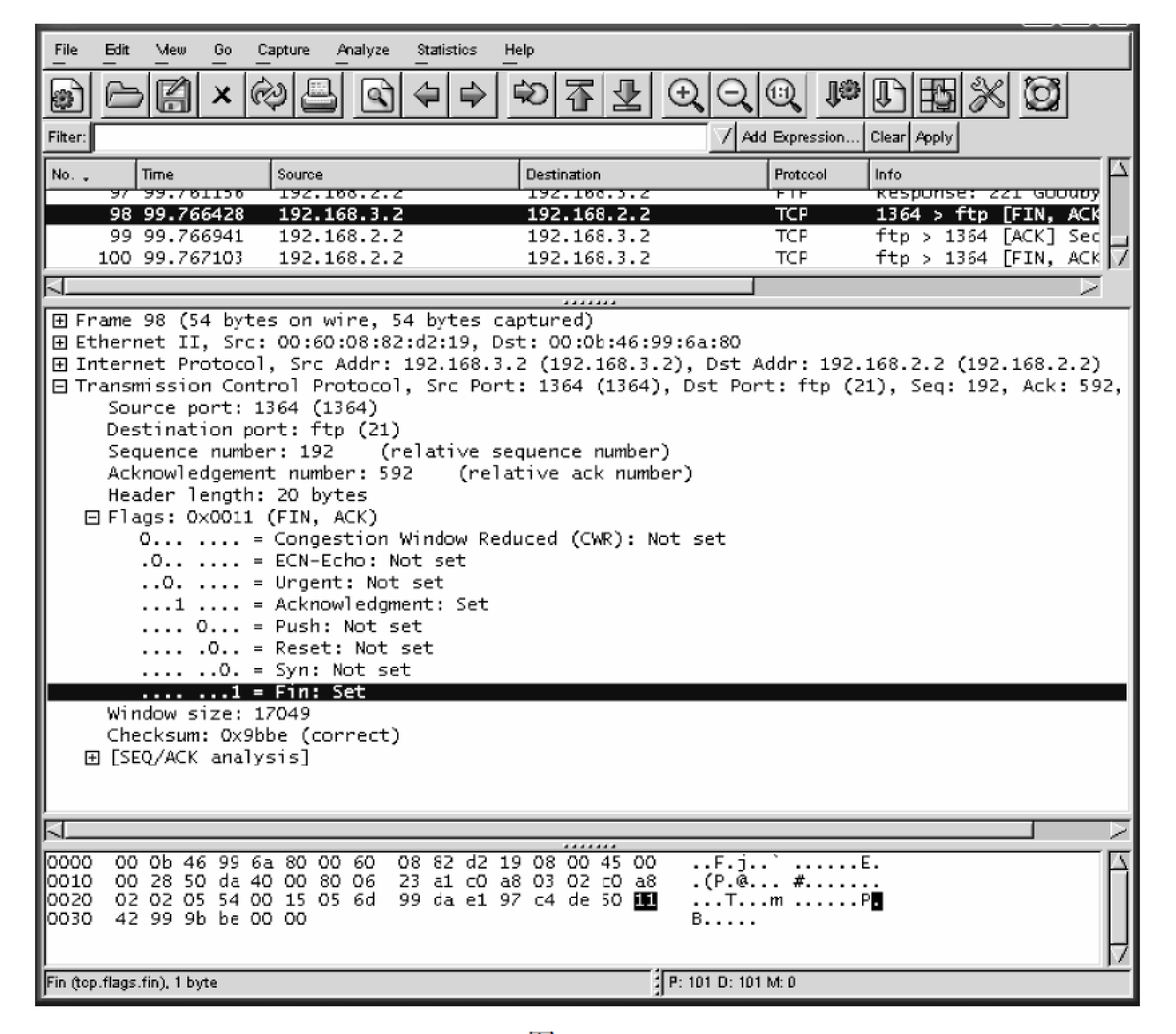

File Edit View Go Capture Analyze Statistics Help
Filter:
Add Expression... Clear Apply
No. Time Source Destination Protocol Info
98 99.766428 192.168.3.2 192.168.2.2 TCP 1364 > ftp [FIN, ACK
99 99.766941 192.168.2.2 192.168.3.2 TCP ftp > 1364 [ACK] Sec
100 99.767103 192.168.2.2 192.168.3.2 TCP ftp > 1364 [FIN, ACK
Frame 98 (54 bytes on wire, 54 bytes captured)
Ethernet II, Src: 00:60:08:82:d2:19, Dst: 00:0b:46:99:6a:80
Internet Protocol, Src Addr: 192.168.3.2 (192.168.3.2), Dst Addr: 192.168.2.2 (192.168.2.2)
Transmission Control Protocol, Src Port: 1364 (1364), Dst Port: ftp (21), Seq: 192, Ack: 592,
Source port: 1364 (1364)
Destination port: ftp (21)
Sequence number: 192 (relative sequence number)
Acknowledgement number: 592 (relative ack number)
Header length: 20 bytes
Flags: 0x0011 (FIN, ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 0... = Push: Not set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...1 = Fin: Set
Window size: 17049
Checksum: 0x9bbe (correct)
[SEQ/ACK analysis]
0000 00 0b 46 99 6a 80 00 60 08 82 d2 19 08 00 45 00 ..F.j..` ......E.
0010 00 28 50 da 40 00 80 06 23 a1 c0 a8 03 02 c0 a8 .(P.@... #.......
0020 02 02 05 54 00 15 05 6d 99 ca e1 97 c4 de 50 11 ...T...m ......P.
0030 42 99 9b be 00 00 B.....
Fin (tcp.flags.fin), 1 byte
P: 101 D: 101 M: 0

图　3-15

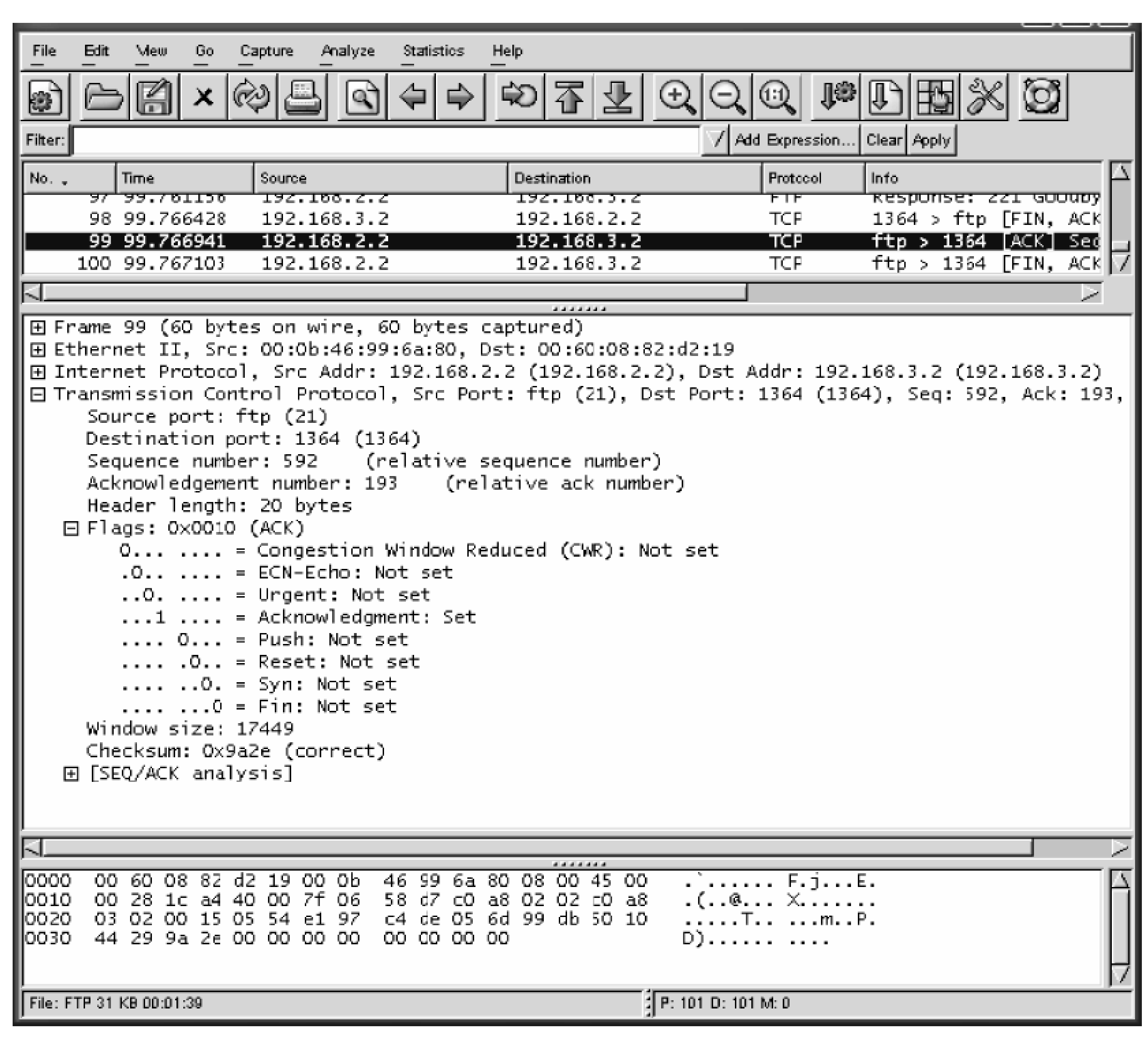

File Edit View Go Capture Analyze Statistics Help
Filter:
Add Expression... Clear Apply
No. Time Source Destination Protocol Info
98 99.766428 192.168.3.2 192.168.2.2 TCP 1364 > ftp [FIN, ACK
99 99.766941 192.168.2.2 192.168.3.2 TCP ftp > 1364 [ACK] Sec
100 99.767103 192.168.2.2 192.168.3.2 TCP ftp > 1364 [FIN, ACK
Frame 99 (60 bytes on wire, 60 bytes captured)
Ethernet II, Src: 00:0b:46:99:6a:80, Dst: 00:60:08:82:d2:19
Internet Protocol, Src Addr: 192.168.2.2 (192.168.2.2), Dst Addr: 192.168.3.2 (192.168.3.2)
Transmission Control Protocol, Src Port: ftp (21), Dst Port: 1364 (1364), Seq: 592, Ack: 193,
Source port: ftp (21)
Destination port: 1364 (1364)
Sequence number: 592 (relative sequence number)
Acknowledgement number: 193 (relative ack number)
Header length: 20 bytes
Flags: 0x0010 (ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 0... = Push: Not set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 17449
Checksum: 0x9a2e (correct)
[SEQ/ACK analysis]
0000 00 60 08 82 d2 19 00 0b 46 99 6a 80 08 00 45 00 .`...... F.j...E.
0010 00 28 1c a4 40 00 7f 06 58 d7 c0 a8 02 02 c0 a8 .(..@... X.......
0020 03 02 00 15 05 54 e1 97 c4 de 05 6d 99 db 50 10 .....T.. ...m..P.
0030 44 29 9a 2e 00 00 00 00 00 00 00 00 D)...... ....
File: FTP 31 KB 00:01:39
P: 101 D: 101 M: 0

图　3-16

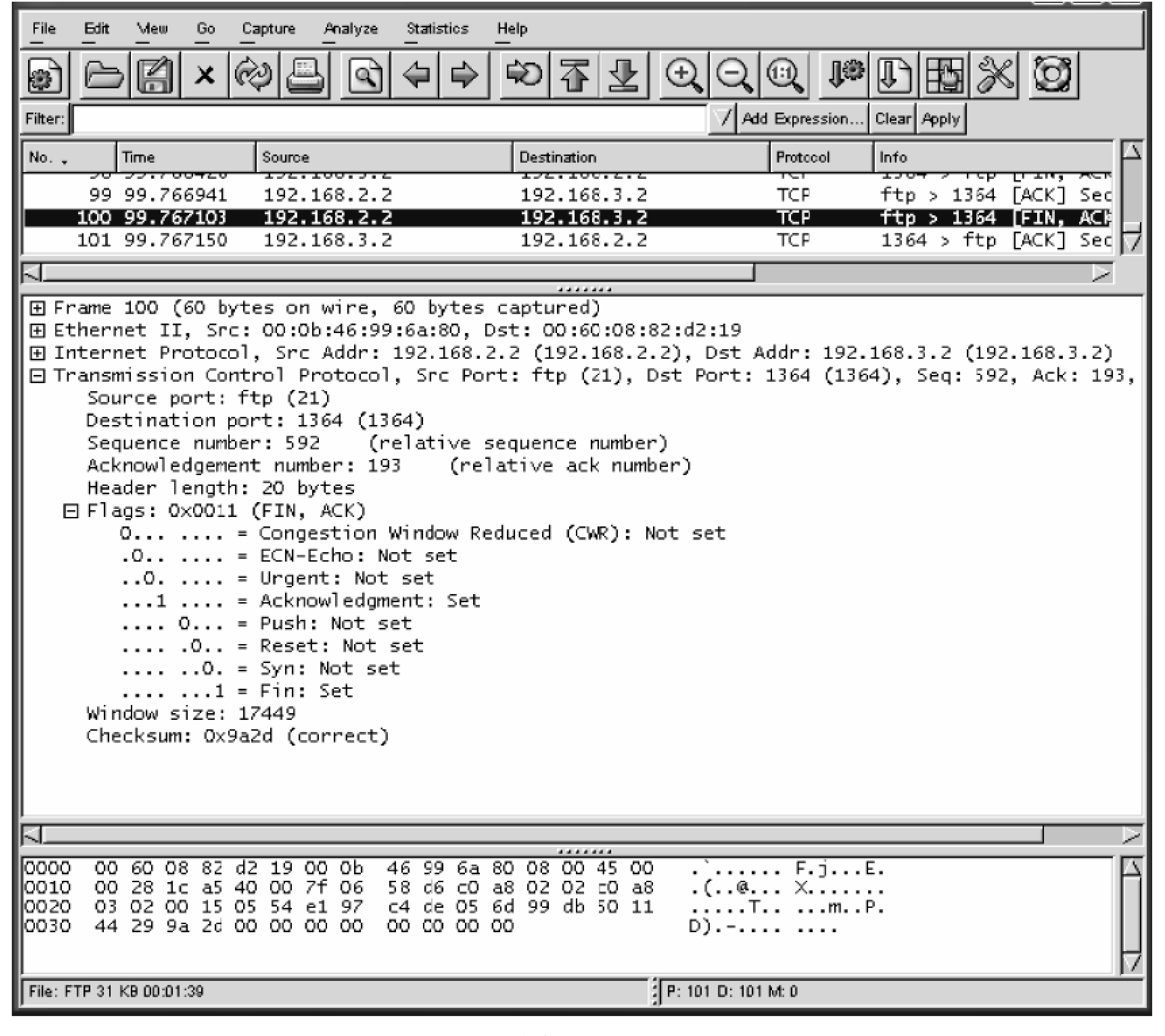

File Edit View Go Capture Analyze Statistics Help
Filter:
Add Expression... Clear Apply
No. Time Source Destination Protocol Info
99 99.766941 192.168.2.2 192.168.3.2 TCP ftp > 1364 [ACK] Sec
100 99.767103 192.168.2.2 192.168.3.2 TCP ftp > 1364 [FIN, ACK
101 99.767150 192.168.3.2 192.168.2.2 TCP 1364 > ftp [ACK] Sec
Frame 100 (60 bytes on wire, 60 bytes captured)
Ethernet II, Src: 00:0b:46:99:6a:80, Dst: 00:60:08:82:d2:19
Internet Protocol, Src Addr: 192.168.2.2 (192.168.2.2), Dst Addr: 192.168.3.2 (192.168.3.2)
Transmission Control Protocol, Src Port: ftp (21), Dst Port: 1364 (1364), Seq: 592, Ack: 193,
Source port: ftp (21)
Destination port: 1364 (1364)
Sequence number: 592 (relative sequence number)
Acknowledgement number: 193 (relative ack number)
Header length: 20 bytes
Flags: 0x0011 (FIN, ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 0... = Push: Not set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...1 = Fin: Set
Window size: 17449
Checksum: 0x9a2d (correct)
0000 00 60 08 82 d2 19 00 0b 46 99 6a 80 08 00 45 00 .`...... F.j...E.
0010 00 28 1c a5 40 00 7f 06 58 c6 c0 a8 02 02 c0 a8 .(..@... X.......
0020 03 02 00 15 05 54 e1 97 c4 ce 05 6d 99 db 50 11 .....T.. ...m..P.
0030 44 29 9a 2d 00 00 00 00 00 00 00 00 D).-.... ....
File: FTP 31 KB 00:01:39
P: 101 D: 101 M: 0

图 3-17

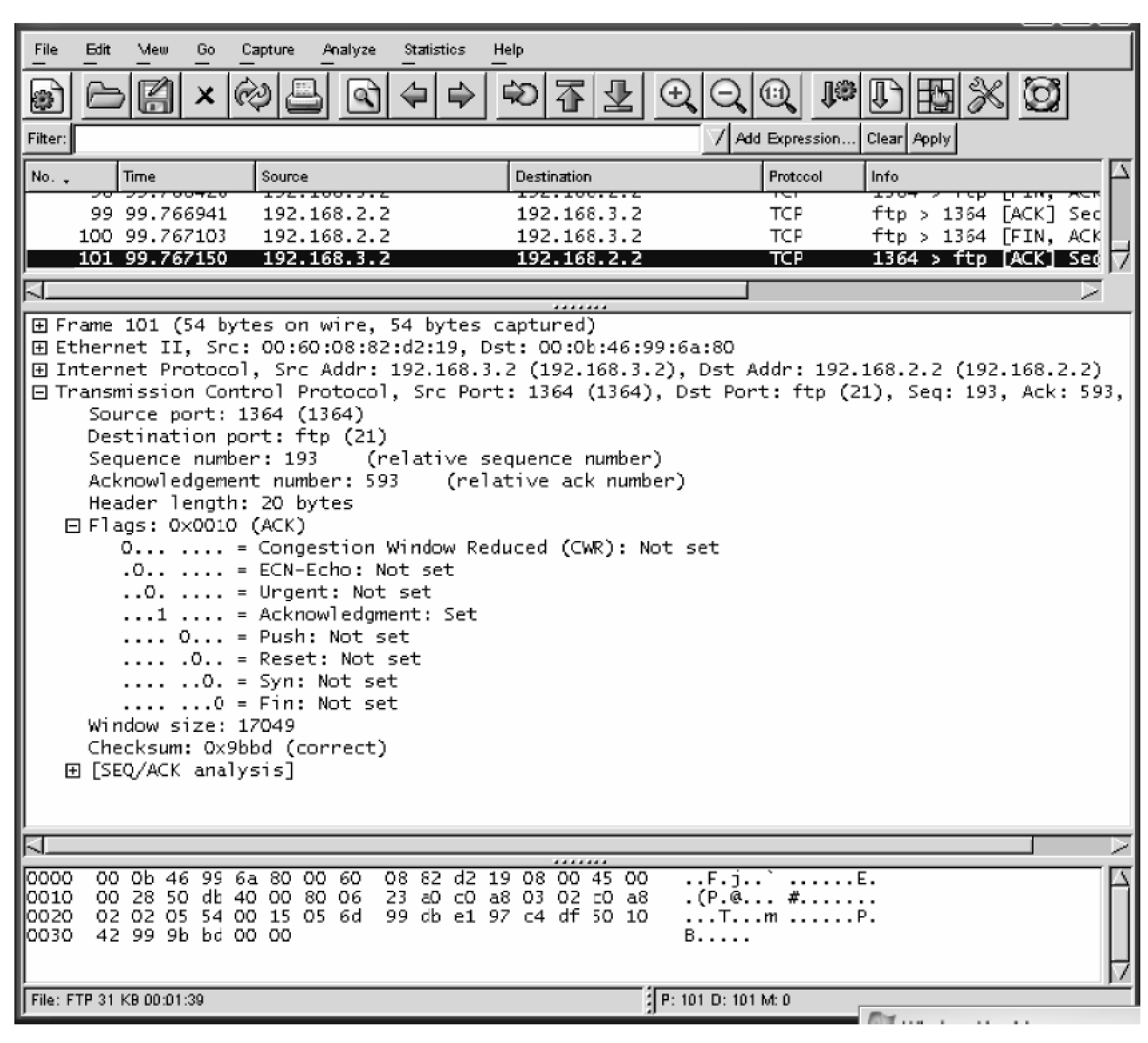

File Edit View Go Capture Analyze Statistics Help
Filter:
Add Expression... Clear Apply
No. Time Source Destination Protocol Info
99 99.766941 192.168.2.2 192.168.3.2 TCP ftp > 1364 [ACK] Sec
100 99.767103 192.168.2.2 192.168.3.2 TCP ftp > 1364 [FIN, ACK
101 99.767150 192.168.3.2 192.168.2.2 TCP 1364 > ftp [ACK] Sec
Frame 101 (54 bytes on wire, 54 bytes captured)
Ethernet II, Src: 00:60:08:82:d2:19, Dst: 00:0b:46:99:6a:80
Internet Protocol, Src Addr: 192.168.3.2 (192.168.3.2), Dst Addr: 192.168.2.2 (192.168.2.2)
Transmission Control Protocol, Src Port: 1364 (1364), Dst Port: ftp (21), Seq: 193, Ack: 593,
Source port: 1364 (1364)
Destination port: ftp (21)
Sequence number: 193 (relative sequence number)
Acknowledgement number: 593 (relative ack number)
Header length: 20 bytes
Flags: 0x0010 (ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 0... = Push: Not set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 17049
Checksum: 0x9bbd (correct)
[SEQ/ACK analysis]
0000 00 0b 46 99 6a 80 00 60 08 82 d2 19 08 00 45 00 ..F.j..` ......E.
0010 00 28 50 db 40 00 80 06 23 a0 c0 a8 03 02 c0 a8 .(P.@... #.......
0020 02 02 05 54 00 15 05 6d 99 cb e1 97 c4 df 50 10 ...T...m ......P.
0030 42 99 9b bd 00 00 B.....
File: FTP 31 KB 00:01:39
P: 101 D: 101 M: 0

图 3-18

## 3.1.6　TCP 传输中的序列号分析

### 1. 三次握手期间的序列号和确认号

如图 3-19 所示，客户端 192.168.3.2 发送 TCP SYN 请求连接，Sequence number 字段为 0，这个值是逻辑值，实际值可以看图 3-19 最下面灰色部分的十六进制值 056d991a，转换成十进制数为 91 068 698。

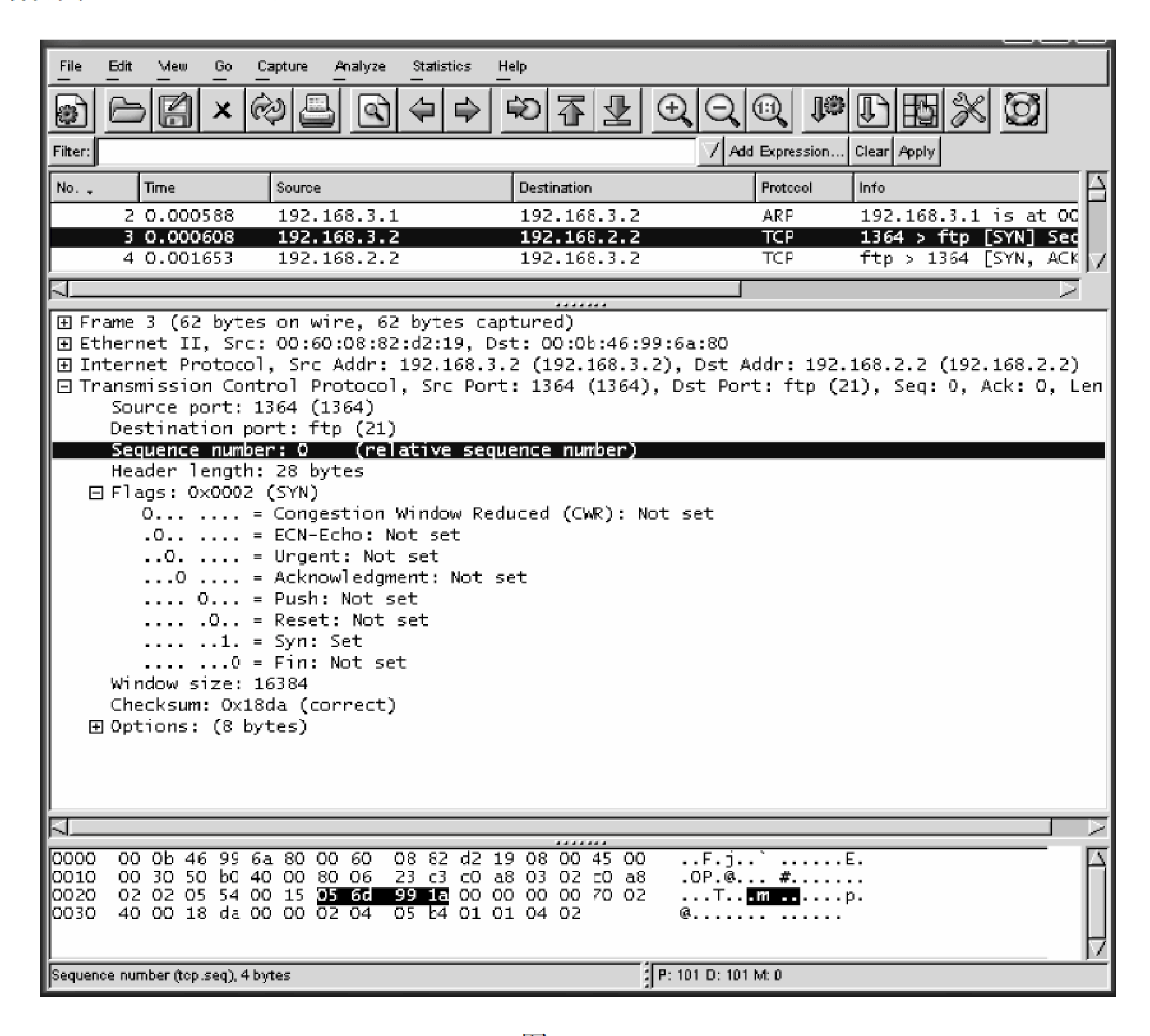



图　3-19

接下来服务器端 192.168.2.2 发送 TCP SYN+ACK，如图 3-20 所示，Sequence number 字段为 0，这个值也是逻辑值，实际值可以看最下面灰色部分的十六进制值 e197c28e，转换成十进制数为 3784819342。Acknowledgement number 字段逻辑值为 1，实际的十六进制值为 056d991b，转换为十进制数为 91 068 699，对前面的数据包的确认即上一个包的 Sequence number+1。

如图 3-21 所示，客户端 192.168.3.2 发送 ACK，Sequence number 逻辑值为 1，十六进制值 056d991b，是客户端发送第一个包的 Sequence number 值+1。Acknowledgement number 逻辑值为 1，十六进制值为 e197c28f，转换为十进制数为 3 784 819 343，是对服务器端发送的包的确认，上一个包的 Sequence number 值为 e197c28e。

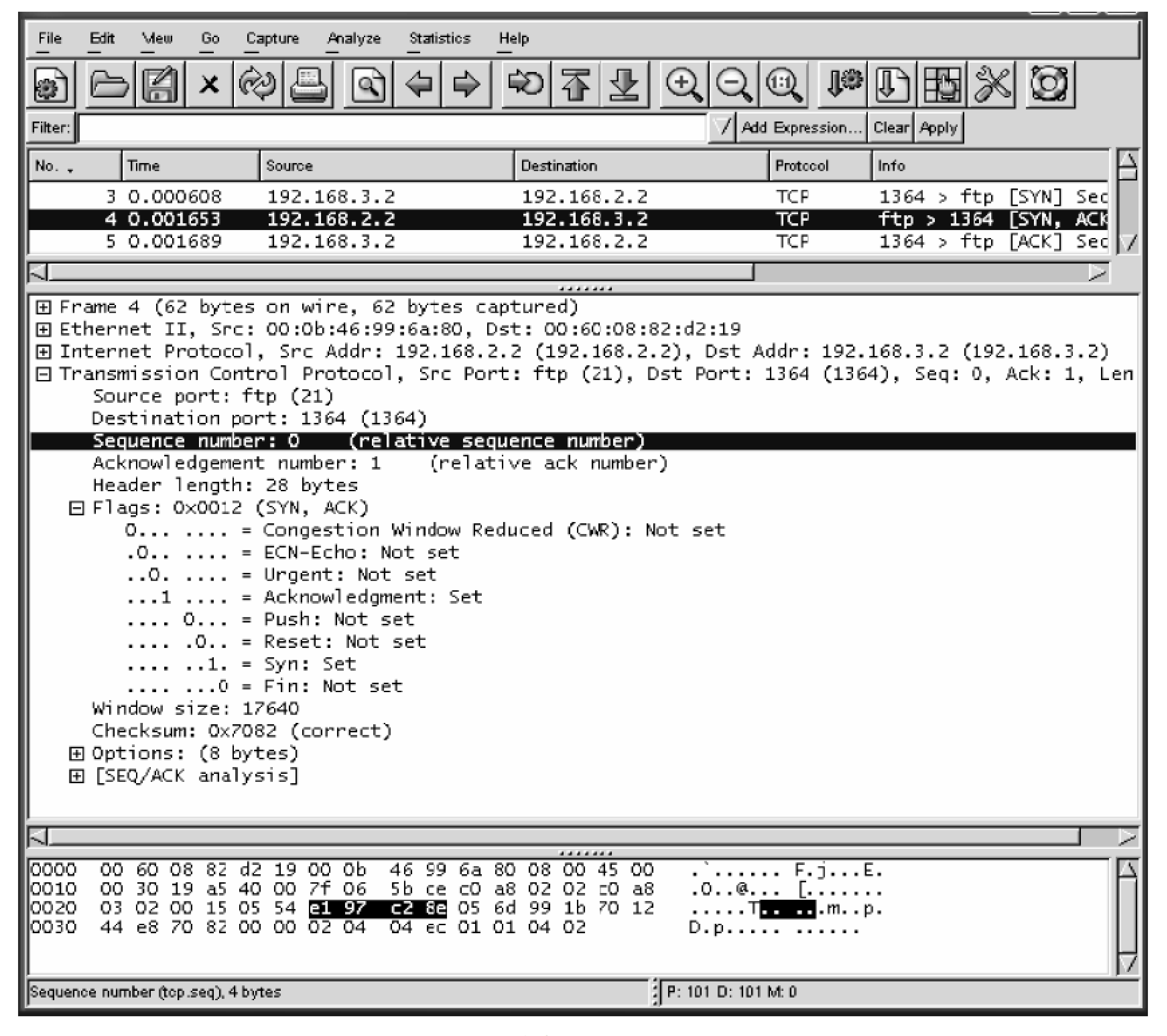

File Edit View Go Capture Analyze Statistics Help
Filter: Add Expression... Clear Apply
No. Time Source Destination Protocol Info
3 0.000608 192.168.3.2 192.168.2.2 TCP 1364 > ftp [SYN] Sec
4 0.001653 192.168.2.2 192.168.3.2 TCP ftp > 1364 [SYN, ACK
5 0.001689 192.168.3.2 192.168.2.2 TCP 1364 > ftp [ACK] Sec
Frame 4 (62 bytes on wire, 62 bytes captured)
Ethernet II, Src: 00:0b:46:99:6a:80, Dst: 00:60:08:82:d2:19
Internet Protocol, Src Addr: 192.168.2.2 (192.168.2.2), Dst Addr: 192.168.3.2 (192.168.3.2)
Transmission Control Protocol, Src Port: ftp (21), Dst Port: 1364 (1364), Seq: 0, Ack: 1, Len
Source port: ftp (21)
Destination port: 1364 (1364)
Sequence number: 0 (relative sequence number)
Acknowledgement number: 1 (relative ack number)
Header length: 28 bytes
Flags: 0x0012 (SYN, ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 0... = Push: Not set
.... .0.. = Reset: Not set
.... ..1. = Syn: Set
.... ...0 = Fin: Not set
Window size: 17640
Checksum: 0x7082 (correct)
Options: (8 bytes)
[SEQ/ACK analysis]
0000 00 60 08 82 d2 19 00 0b 46 99 6a 80 08 00 45 00
0010 00 30 19 a5 40 00 7f 06 5b ce c0 a8 02 02 c0 a8
0020 03 02 00 15 05 54 e1 97 c2 8e 05 6d 99 1b 70 12
0030 44 e8 70 82 00 00 02 04 04 ec 01 01 04 02
Sequence number (tcp.seq), 4 bytes
P: 101 D: 101 M: 0

图 3-20

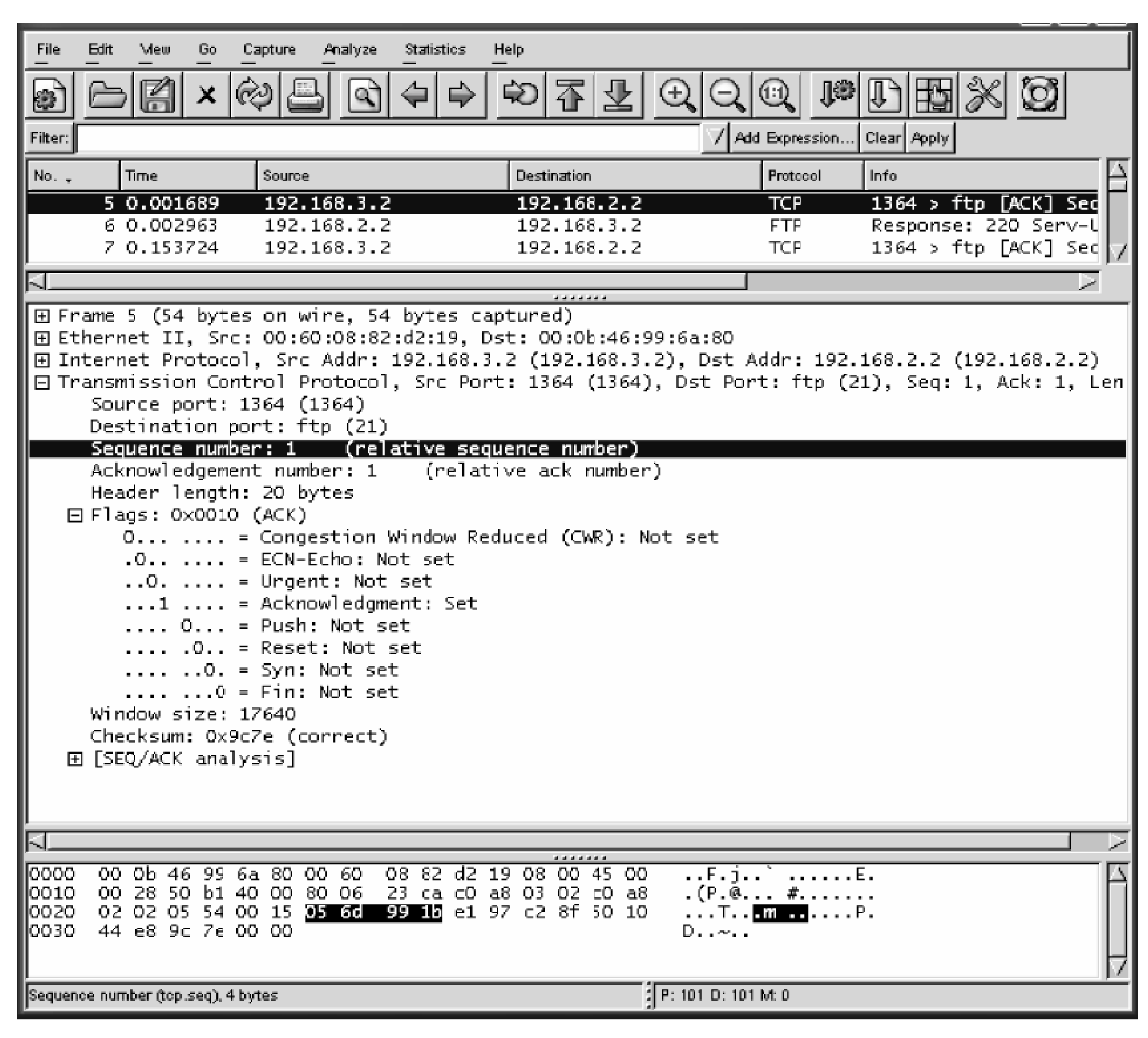

File Edit View Go Capture Analyze Statistics Help
Filter: Add Expression... Clear Apply
No. Time Source Destination Protocol Info
5 0.001689 192.168.3.2 192.168.2.2 TCP 1364 > ftp [ACK] Sec
6 0.002963 192.168.2.2 192.168.3.2 FTP Response: 220 Serv-U
7 0.153724 192.168.3.2 192.168.2.2 TCP 1364 > ftp [ACK] Sec
Frame 5 (54 bytes on wire, 54 bytes captured)
Ethernet II, Src: 00:60:08:82:d2:19, Dst: 00:0b:46:99:6a:80
Internet Protocol, Src Addr: 192.168.3.2 (192.168.3.2), Dst Addr: 192.168.2.2 (192.168.2.2)
Transmission Control Protocol, Src Port: 1364 (1364), Dst Port: ftp (21), Seq: 1, Ack: 1, Len
Source port: 1364 (1364)
Destination port: ftp (21)
Sequence number: 1 (relative sequence number)
Acknowledgement number: 1 (relative ack number)
Header length: 20 bytes
Flags: 0x0010 (ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 0... = Push: Not set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 17640
Checksum: 0x9c7e (correct)
[SEQ/ACK analysis]
0000 00 0b 46 99 6a 80 00 60 08 82 d2 19 08 00 45 00
0010 00 28 50 b1 40 00 80 06 23 ca c0 a8 03 02 c0 a8
0020 02 02 05 54 00 15 05 6d 99 1b e1 97 c2 8f 50 10
0030 44 e8 9c 7e 00 00
Sequence number (tcp.seq), 4 bytes
P: 101 D: 101 M: 0

图 3-21

### 2．传输数据时的序列号和确认号

图 3-22 开始传输数据，Sequence number 的十六进制值为 e1c1edc3，转换十进制数为 3 787 582 915，Acknowledgement number 的十六进制值为 05980a88。逻辑 Sequence number 为 1，表示发送数据段的第一个字节序号为 1，逻辑上 next sequence number 为 59，表示下一个传输字段从第 59 个字节开始传输，这次总共传送 58 个字节。

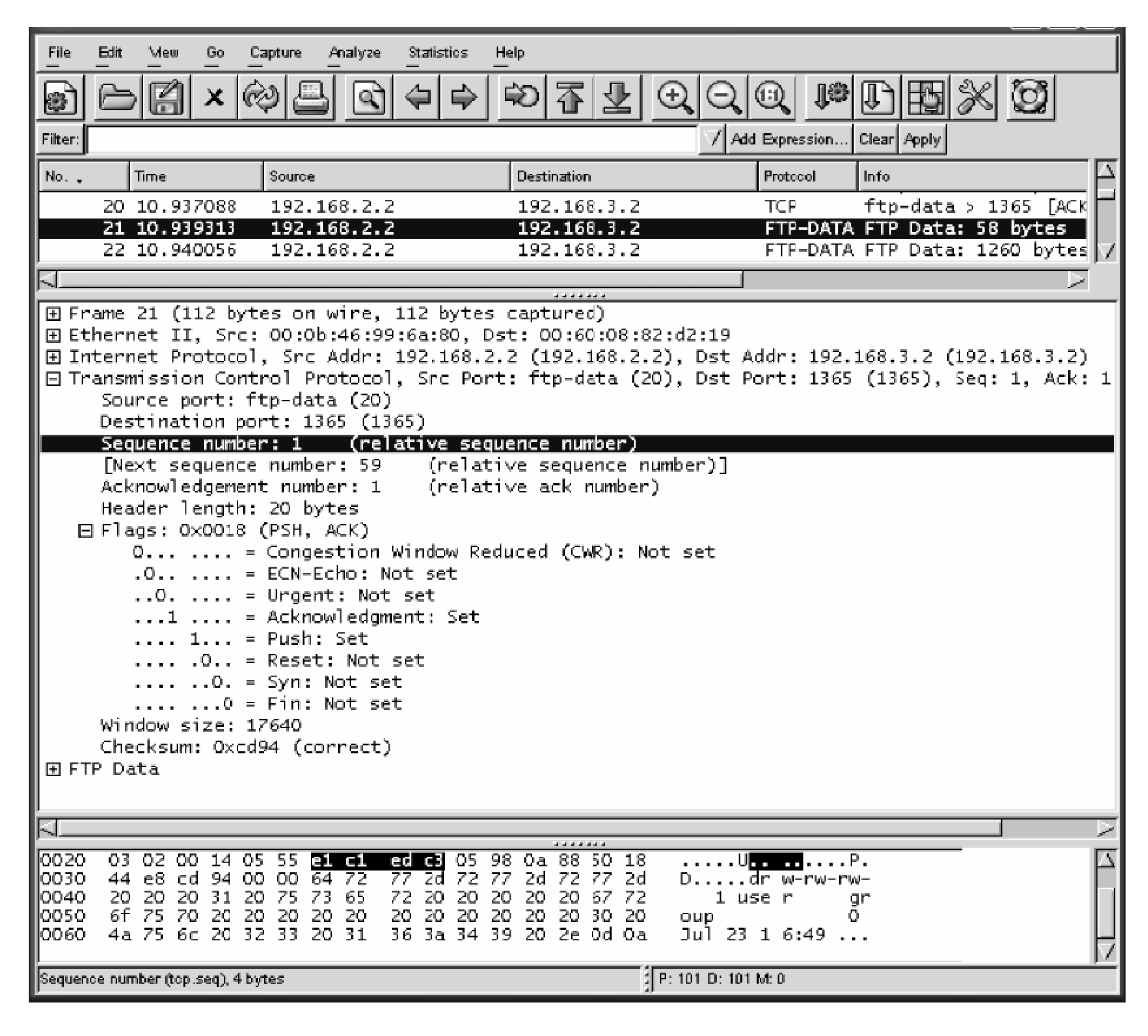


图　3-22

图 3-23 中，继续发送数据，这个数据包的 Sequence number 的十六进制值为 e1c1edfd，转换为十进制数为 3 787 582 973，减去上面的 Sequence number 差为 58，表示传输了 58 个字节，可以看到这个数据包的 Sequence number 逻辑值是 59，即 1＋58＝59，逻辑上 Next Sequence number 值为 1319，表示下一个传输字段从第 1319 个字节开始传输，这次总共传输 1260 个字节。

图 3-24 中，继续发送数据，Sequence number 逻辑值是 1319，逻辑上 Next sequence number 值为 1381，表示下一个传输字段从第 1381 个字节开始传输，这次总共传输的 62 个字节。

图 3-25 中 Acknowledgement number 为 1381，表示收到了 1380 个字节。这里的 1381 数据是逻辑上的，如果有兴趣的话，可以用真实的十六进制数来减去前面的序号，得出的结论是一样的。是对前面三段数据的确认 58＋1260＋62＋1＝1381，TCP 在数据传输中，不用对每段数据进行确认，可以用一个确认号来确认前面收到的所有数据。

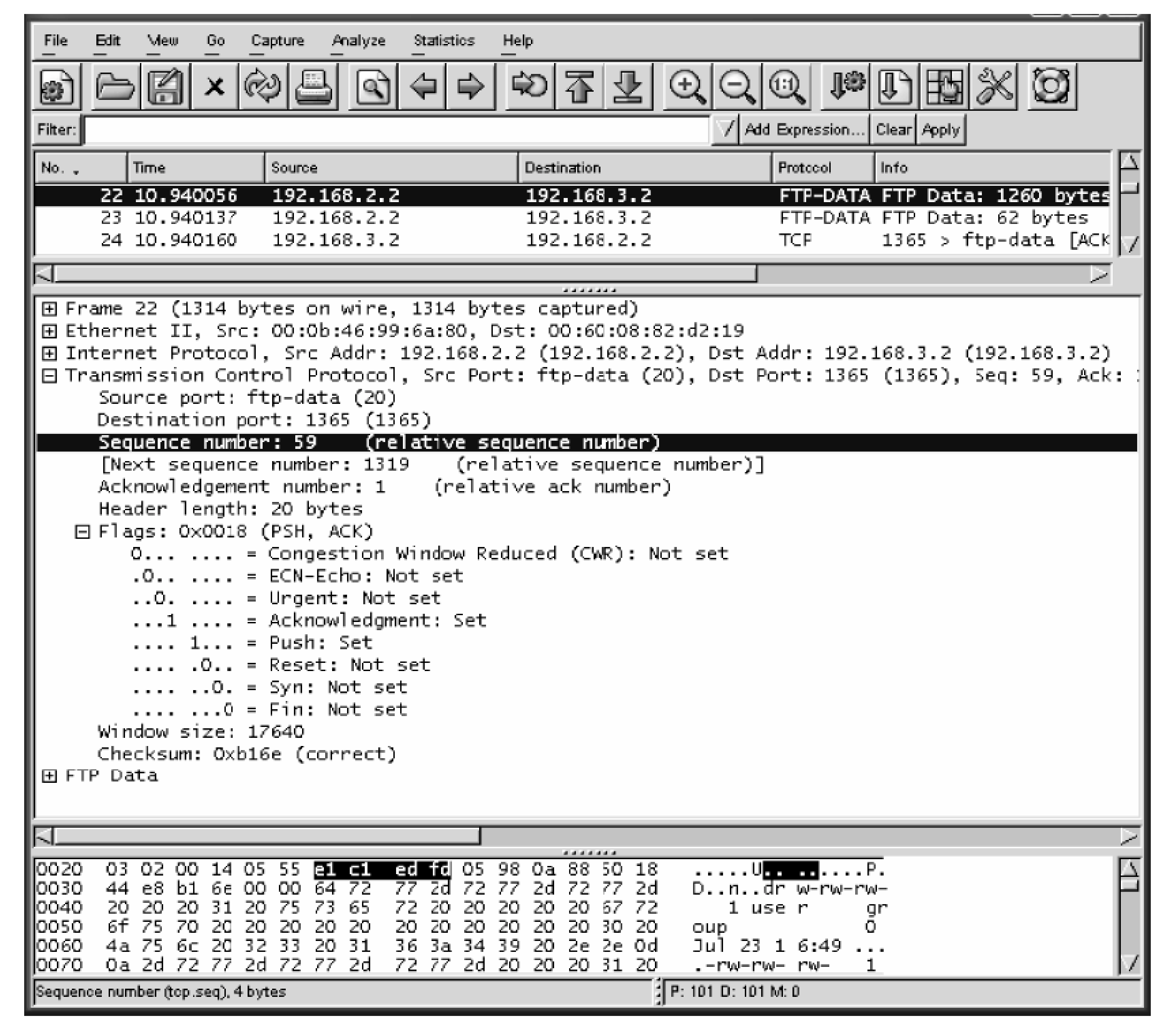

File Edit View Go Capture Analyze Statistics Help
Filter: Add Expression... Clear Apply
No. Time Source Destination Protocol Info
22 10.940056 192.168.2.2 192.168.3.2 FTP-DATA FTP Data: 1260 bytes
23 10.940137 192.168.2.2 192.168.3.2 FTP-DATA FTP Data: 62 bytes
24 10.940160 192.168.3.2 192.168.2.2 TCP 1365 > ftp-data [ACK
Frame 22 (1314 bytes on wire, 1314 bytes captured)
Ethernet II, Src: 00:0b:46:99:6a:80, Dst: 00:60:08:82:d2:19
Internet Protocol, Src Addr: 192.168.2.2 (192.168.2.2), Dst Addr: 192.168.3.2 (192.168.3.2)
Transmission Control Protocol, Src Port: ftp-data (20), Dst Port: 1365 (1365), Seq: 59, Ack:
Source port: ftp-data (20)
Destination port: 1365 (1365)
Sequence number: 59 (relative sequence number)
[Next sequence number: 1319 (relative sequence number)]
Acknowledgement number: 1 (relative ack number)
Header length: 20 bytes
Flags: 0x0018 (PSH, ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 1... = Push: Set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 17640
Checksum: 0xb16e (correct)
FTP Data
0020 03 02 00 14 05 55 e1 c1 ed fd 05 98 0a 88 50 18 .....U.......P.
0030 44 e8 b1 6e 00 00 64 72 77 2d 72 77 2d 72 77 2d D..n..dr w-rw-rw-
0040 20 20 20 31 20 75 73 65 72 20 20 20 20 20 67 72 1 use r gr
0050 6f 75 70 20 20 20 20 20 20 20 20 20 20 20 30 20 oup 0
0060 4a 75 6c 20 32 33 20 31 36 3a 34 39 20 2e 2e 0d Jul 23 1 6:49 ...
0070 0a 2d 72 77 2d 72 77 2d 72 77 2d 20 20 20 31 20 .-rw-rw- rw- 1
Sequence number (tcp.seq), 4 bytes
P: 101 D: 101 M: 0

图 3-23

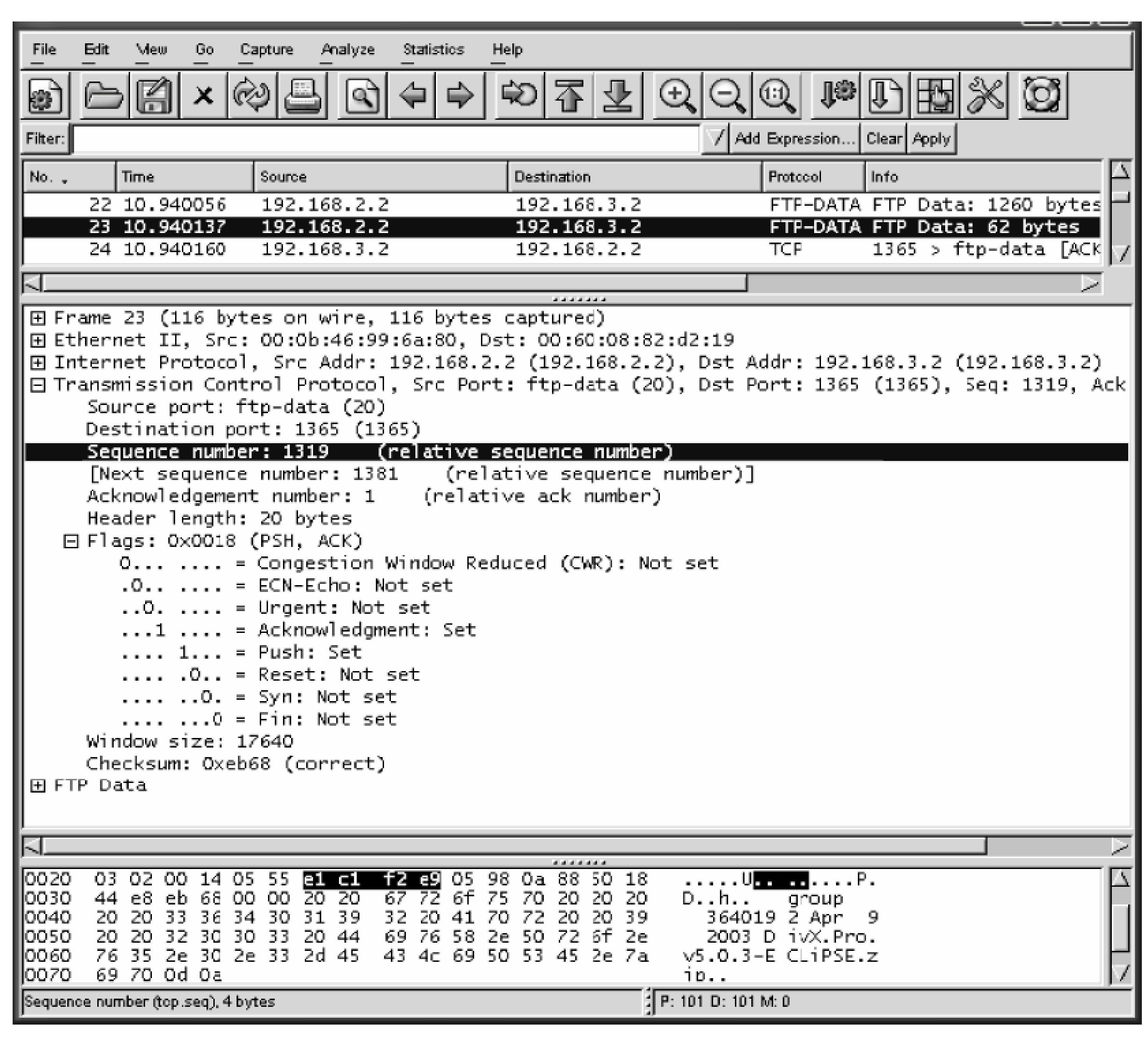

File Edit View Go Capture Analyze Statistics Help
Filter: Add Expression... Clear Apply
No. Time Source Destination Protocol Info
22 10.940056 192.168.2.2 192.168.3.2 FTP-DATA FTP Data: 1260 bytes
23 10.940137 192.168.2.2 192.168.3.2 FTP-DATA FTP Data: 62 bytes
24 10.940160 192.168.3.2 192.168.2.2 TCP 1365 > ftp-data [ACK
Frame 23 (116 bytes on wire, 116 bytes captured)
Ethernet II, Src: 00:0b:46:99:6a:80, Dst: 00:60:08:82:d2:19
Internet Protocol, Src Addr: 192.168.2.2 (192.168.2.2), Dst Addr: 192.168.3.2 (192.168.3.2)
Transmission Control Protocol, Src Port: ftp-data (20), Dst Port: 1365 (1365), Seq: 1319, Ack
Source port: ftp-data (20)
Destination port: 1365 (1365)
Sequence number: 1319 (relative sequence number)
[Next sequence number: 1381 (relative sequence number)]
Acknowledgement number: 1 (relative ack number)
Header length: 20 bytes
Flags: 0x0018 (PSH, ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 1... = Push: Set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 17640
Checksum: 0xeb68 (correct)
FTP Data
0020 03 02 00 14 05 55 e1 c1 f2 e9 05 98 0a 88 50 18 .....U.......P.
0030 44 e8 eb 68 00 00 20 20 67 72 6f 75 70 20 20 20 D..h.. group
0040 20 20 33 36 34 30 31 39 32 20 41 70 72 20 20 39 364019 2 Apr 9
0050 20 20 32 30 30 33 20 44 69 76 58 2e 50 72 6f 2e 2003 D ivX.Pro.
0060 76 35 2e 30 2e 33 2d 45 43 4c 69 50 53 45 2e 7a v5.0.3-E CLiPSE.z
0070 69 70 0d 0a ip..
Sequence number (tcp.seq), 4 bytes
P: 101 D: 101 M: 0

图 3-24

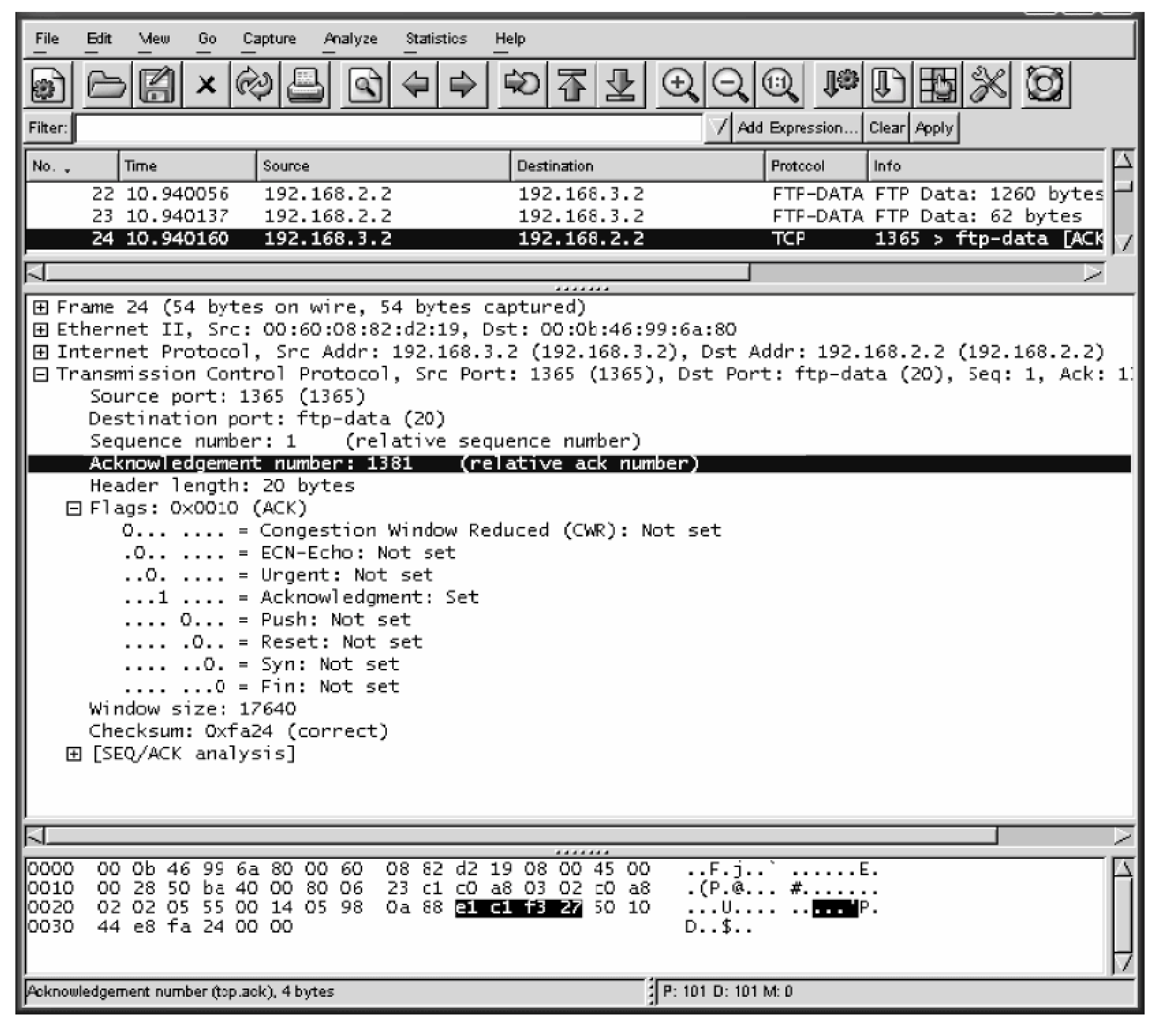



图　3-25

## 3.2　UDP 协议

用户数据包协议(User Datagram Protocol,UDP)主要用来支持那些需要在计算机之间传输数据的网络应用,包括网络视频会议系统在内的众多的客户/服务器模式的网络应用都需要使用 UDP 协议。UDP 协议直接位于 IP 协议的上层。

前面学过 TCP 协议,它有一系列数据传送的保证机制,比如超时重传、每发送一个报文段都有一个序列号等,来保证数据安全到达,不丢失,不失序,然而,UDP 就没有这样的保证机制。当计算机之间利用 UDP 协议传送数据的时候,由于协议本身不能做出任何检测或提示,所以发送方只管发送数据,而并不确认数据是否被对方接收,因此就会导致某些 UDP 协议数据包在传送的过程中丢失,尤其网络质量不令人满意的情况下,丢失数据包的现象会更严重。因此,通常人们把 UDP 协议称为不可靠的传输协议。

那么既然 UDP 是一种不可靠的网络协议,为什么还要使用它呢?虽然 TCP 协议中植入了各种安全保障功能,但是在实际执行的过程中会占用大量的系统开销,无疑使速度受到严重的影响。UDP 由于排除了信息可靠传递机制,将安全和排序等功能移交给上层应用来完成,极大降低了执行时间,使速度得到了保证。

关于 UDP 协议的最早规范是 1980 年发布的 RFC768。UDP 协议在主流应用中发挥着重要作用。

流媒体、实时多媒体游戏和 IP 电话 (VoIP)就是典型的 UDP 应用。因为相对于可靠性来说,这些应用更加注重实际性能,所以为了获得更好的使用效果(例如,更高的画面帧刷新

速率)往往可以牺牲一定的可靠性(例如,会面质量)。这就是 UDP 和 TCP 两种协议的权衡之处。根据不同的环境和特点,两种传输协议都将在今后的网络世界中发挥更加重要的作用。

UDP 报文的格式如图 3-26 所示。

| | |
|---|---|
| 16 位源端口号 | 16 位目的端口号 |
| 16 位长度 | 16 位校验和 |
| 数据 | |

图 3-26

每个 UDP 报文分为两部分:头部和数据区。报文头中包含 16 位源端口号,16 位目的端口号,16 位 UDP 长度(首部和数据),16 位检验和,下面分别来解释这 6 个字段的含义和作用。

源端口(Source port)和目的端口(Destination port)字段包含了 16 位的 UDP 协议端口号,UDP 协议与 TCP 协议一样都用源端口号来标明该分段是由本地进程的哪一个进程创建的,用目的端口号来标明所请求的服务类型(目的端的哪一个应用程序来响应该请求)。通常用的 UDP 的端口号有 DNS 53、BOOTP 67(SERVER)/68(CLIENT)、TFTP 69 和 SNMP 161 等。

数据包的长度是指包括报头和数据部分在内的总的字节数。因为报头的长度是固定的,所以该域主要被用来计算可变长度的数据部分。数据包的最大长度根据操作环境的不同而不同。从理论上说,包含报头在内的数据包的最大长度为 65 535 字节。不过一些实际应用往往会限制数据包的大小,有时会降低到 8192 字节。

UDP 协议使用报头中的校验和来保证数据的安全。校验和首先在数据发送方通过特殊的算法计算得出,在传递到接收方之后,还需要再重新计算。如果某个数据包在传输过程中被第三方篡改或者由于线路噪音等原因受到损坏,发送和接收方的校验计算值将不会相符,因此 UDP 协议可以检测是否出错。在 UDP 协议中校验功能是可选的,如果将其关闭可以使系统的性能有所增强。这与 TCP 协议是不同的,TCP 要求必须具有校验和。

图 3-27 是实际捕获的一个 UDP 包信息,在这个图中可以看到一个 UDP 报文的包头信息。

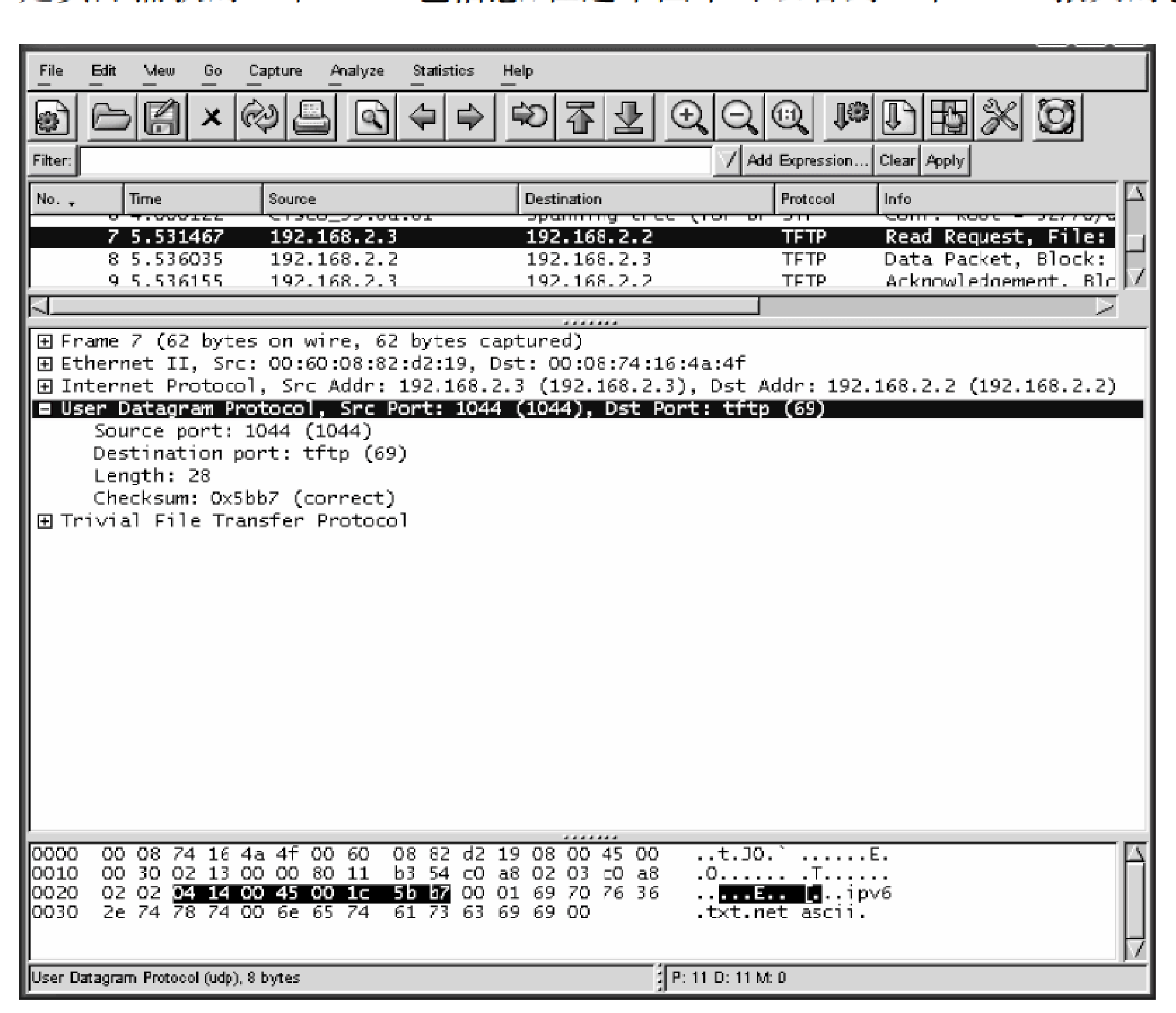



图 3-27

CHAPTER 4

# 第4章 ARP协议

## 4.1 ARP工作原理

ARP协议负责把IP地址转换成网卡硬件地址。

ARP完成的这个任务过程是：发送一个广播，这个广播包含了一个指定的IP地址，要求正在使用这个IP地址的主机响应它的硬件地址。使用这个IP的主机侦听到这个广播（局域网中其他的设备也会侦听到），就会给源端反馈一个ARP响应。注意这个响应不再是网络上的广播，而是直接发送给那个发出请求的主机。

ARP包直接与数据链路层通信，这和IP包是一样的，但是ARP包和IP包是完全独立的，ARP的协议ID是0806，而IP使用的是0800，如图4-1和图4-2所示，可以看到ARP和IP都是直接位于数据链路层的上层，在数据链路层中有个TYPE字段，指明上层协议的类型，可以看到ARP为0x0806，IP为0x0800。

ARP包含几个字段，但是其中只有5个字段是用来提供ARP的整体功能：源端的硬件地址，源端的IP地址，目标的硬件地址，目标的IP地址和一个“消息类型”字段，这个消息类型字段用来表明当前ARP包是一个请求还是对请求的一个响应。

当一个设备发送ARP请求时，它要填充4个与地址相关的字段中的3个，即提供自身的硬件地址和IP地址，还提供目标的IP地址（因为目标的硬件地址是不知道的，所以这个字段填充为0）。另外，它会设置消息类型字段来表明当前包是一个ARP包，然后在局域网上广播这个请求，使得所有设备都能侦听到。一旦主机侦听到针对自己的请求（在ARP请求的IP地址字段中表明），就生成一个响应报文并反馈给请求主机。响应报文包含本地设备的IP地址和硬件地址，还包含源发送系统的IP地址和硬件地址，这个响应报文会通过消息类型字段来表明当前的报文是一个ARP响应报文，新的ARP响应报文不再是广播发送，会直接传输给源端请求者。

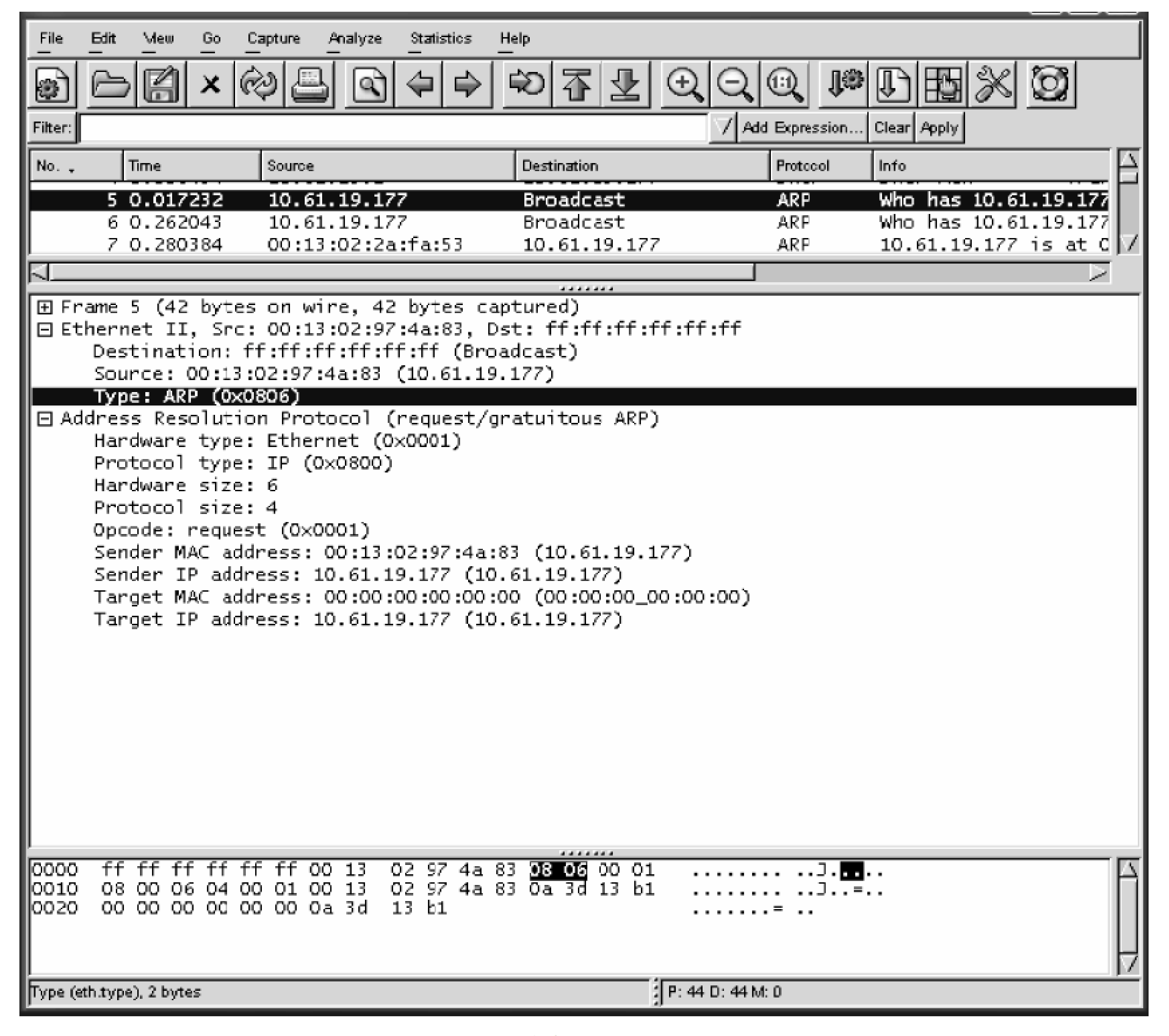

File Edit View Go Capture Analyze Statistics Help
Filter:
Add Expression... Clear Apply
No. Time Source Destination Protocol Info
5 0.017232 10.61.19.177 Broadcast ARP Who has 10.61.19.177
6 0.262043 10.61.19.177 Broadcast ARP Who has 10.61.19.177
7 0.280384 00:13:02:2a:fa:53 10.61.19.177 ARP 10.61.19.177 is at 0
Frame 5 (42 bytes on wire, 42 bytes captured)
Ethernet II, Src: 00:13:02:97:4a:83, Dst: ff:ff:ff:ff:ff:ff
Destination: ff:ff:ff:ff:ff:ff (Broadcast)
Source: 00:13:02:97:4a:83 (10.61.19.177)
Type: ARP (0x0806)
Address Resolution Protocol (request/gratuitous ARP)
Hardware type: Ethernet (0x0001)
Protocol type: IP (0x0800)
Hardware size: 6
Protocol size: 4
Opcode: request (0x0001)
Sender MAC address: 00:13:02:97:4a:83 (10.61.19.177)
Sender IP address: 10.61.19.177 (10.61.19.177)
Target MAC address: 00:00:00:00:00:00 (00:00:00_00:00:00)
Target IP address: 10.61.19.177 (10.61.19.177)
0000 ff ff ff ff ff ff 00 13 02 97 4a 83 08 06 00 01 ........ ..J.....
0010 08 00 06 04 00 01 00 13 02 97 4a 83 0a 3d 13 b1 ........ ..J..=..
0020 00 00 00 00 00 00 0a 3d 13 b1 .......= ..
Type (eth.type), 2 bytes
P: 44 D: 44 M: 0

图 4-1

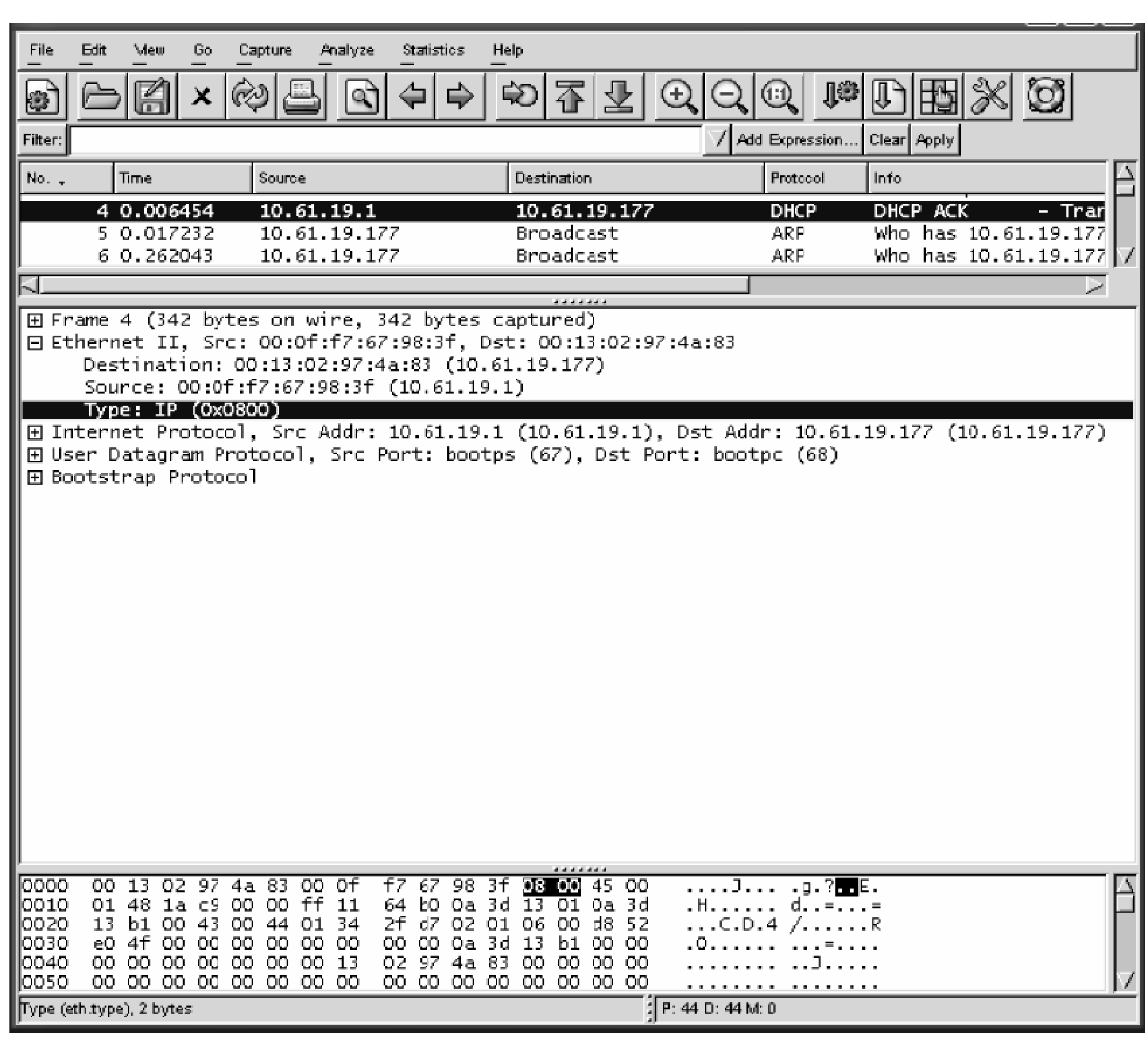

File Edit View Go Capture Analyze Statistics Help
Filter:
Add Expression... Clear Apply
No. Time Source Destination Protocol Info
4 0.006454 10.61.19.1 10.61.19.177 DHCP DHCP ACK - Tran
5 0.017232 10.61.19.177 Broadcast ARP Who has 10.61.19.177
6 0.262043 10.61.19.177 Broadcast ARP Who has 10.61.19.177
Frame 4 (342 bytes on wire, 342 bytes captured)
Ethernet II, Src: 00:0f:f7:67:98:3f, Dst: 00:13:02:97:4a:83
Destination: 00:13:02:97:4a:83 (10.61.19.177)
Source: 00:0f:f7:67:98:3f (10.61.19.1)
Type: IP (0x0800)
Internet Protocol, Src Addr: 10.61.19.1 (10.61.19.1), Dst Addr: 10.61.19.177 (10.61.19.177)
User Datagram Protocol, Src Port: bootps (67), Dst Port: bootpc (68)
Bootstrap Protocol
0000 00 13 02 97 4a 83 00 0f f7 67 98 3f 08 00 45 00 ....J... .g.?..E.
0010 01 48 1a c9 00 00 ff 11 64 b0 0a 3d 13 01 0a 3d .H...... d..=...=
0020 13 b1 00 43 00 44 01 34 2f c7 02 01 06 00 d8 52 ...C.D.4 /......R
0030 e0 4f 00 00 00 00 00 00 00 00 0a 3d 13 b1 00 00 .O...... ...=....
0040 00 00 00 00 00 00 00 13 02 97 4a 83 00 00 00 00 ........ ..J.....
0050 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ........ ........
Type (eth.type), 2 bytes
P: 44 D: 44 M: 0

图 4-2

# 4.2　ARP 报文结构

ARP 报文由 9 个字段组成，包的总大小根据本地网络媒介上使用的物理地址的大小的变化而变化。表 4-1 是 ARP 报文中的字段。

表　4-1

| 字　　段 | 字节数 | 说　　明 |
|---|---|---|
| Hardware Type | 2 | 标识 ARP 实现在哪种类型的网络上 |
| Protocol Type | 2 | 标识讨论的上层协议信息 |
| Hardware Address Length | 1 | 指定物理媒介的硬件地址的大小，以字节为单位。因为每个网络使用的物理寻址机制不同，所以 ARP 包中的其他字段需要这个字段 |
| Protocol Address Length | 1 | 指定上层协议地址的大小，以字节为单位。IP 的地址总是 4 |
| Message Type | 2 | 标识这个 ARP 包的类型（“请求”或“响应”） |
| Source Hardware Address | 可变 | 标识发送 ARP 广播的主机的硬件地址 |
| Source IP Address | 可变 | 标识发送 ARP 广播的主机的 IP 地址 |
| Destination Hardware Address | 可变 | 标识响应 ARP 广播的主机的硬件地址 |
| Destination IP Address | 可变 | 标识响应 ARP 广播的主机的 IP 地址 |

根据上面的字段，下面用实际截获的 ARP 报文来逐个分析 ARP 报文中的字段。

### 1. Hardware Type

标识被请求的硬件地址类型。DIX-Ethernet 的地址类型是 1，IEEE 802.X Ethernet 的地址类型是 6，ARCnet 的地址类型是 7，如图 4-3 所示。

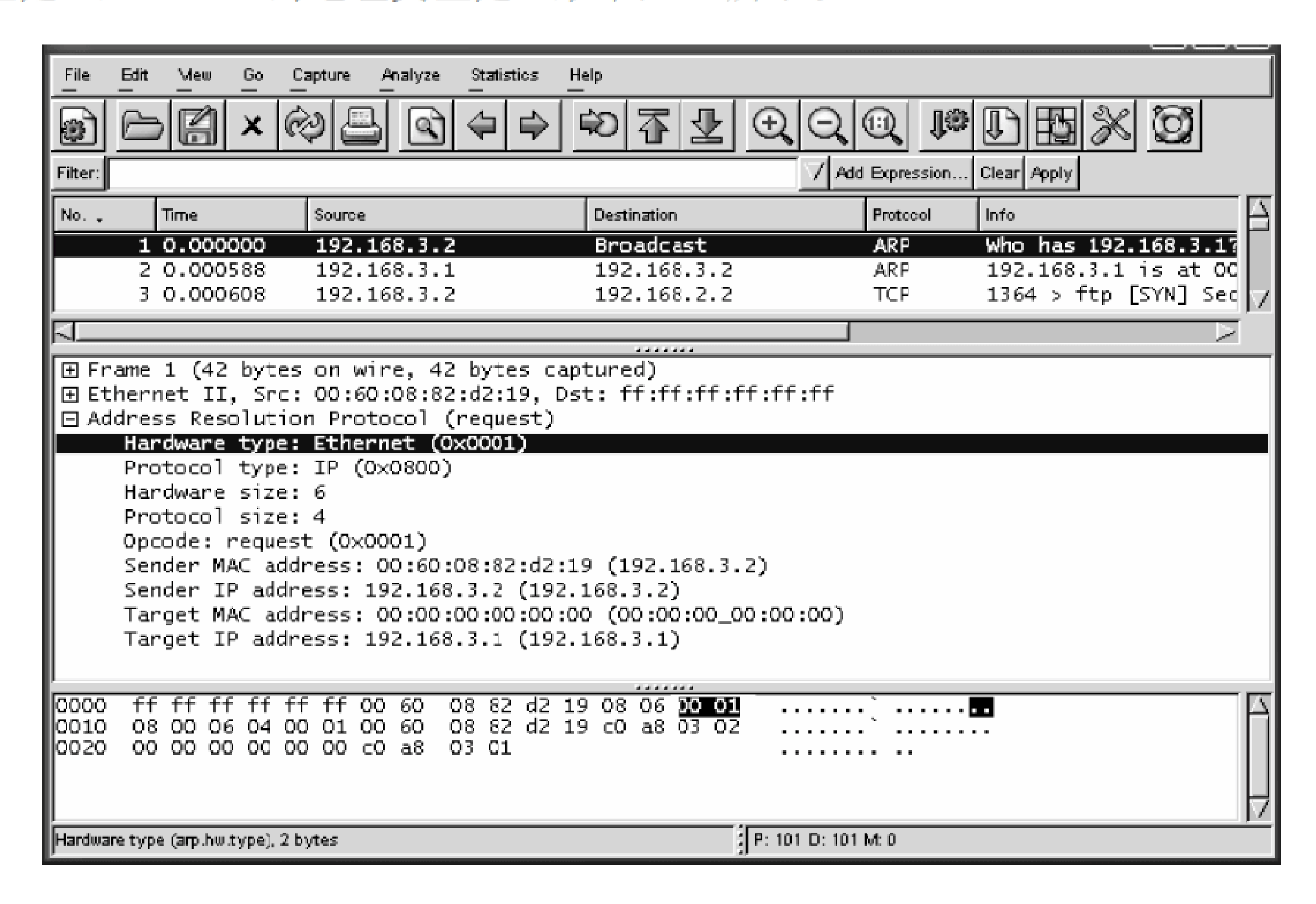



图　4-3

## 2. Protocol Type

标识使用的高层协议。因为 ARP 直接与物理媒介的数据链路的接口通信，所以它并不专用于 IP，而是可以让任何高层协议用来寻找与高层协议地址相关的硬件地址。因此，Protocol Type 字段必须定义这个请求所指的高层协议，如图 4-4 所示。

对于 Ethernet 网络，IP 的值是十六进制的 0800（这与十进制的 2048 是相等的）。

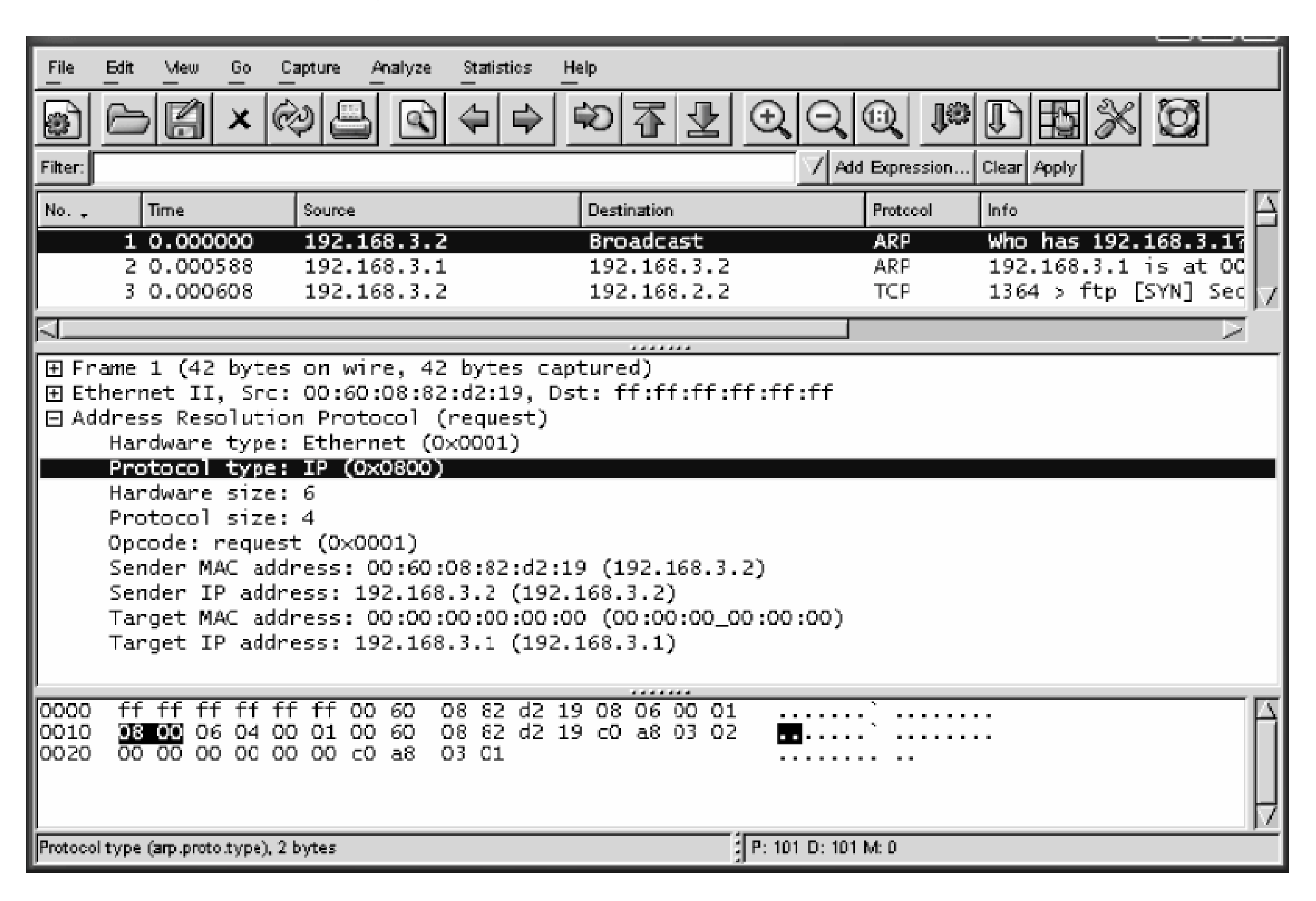



图 4-4

## 3. Hardware Address Length

指定 Source 和 Destination Hardware Address 字段中提供的硬件地址长度（以字节表示），如图 4-5 所示。

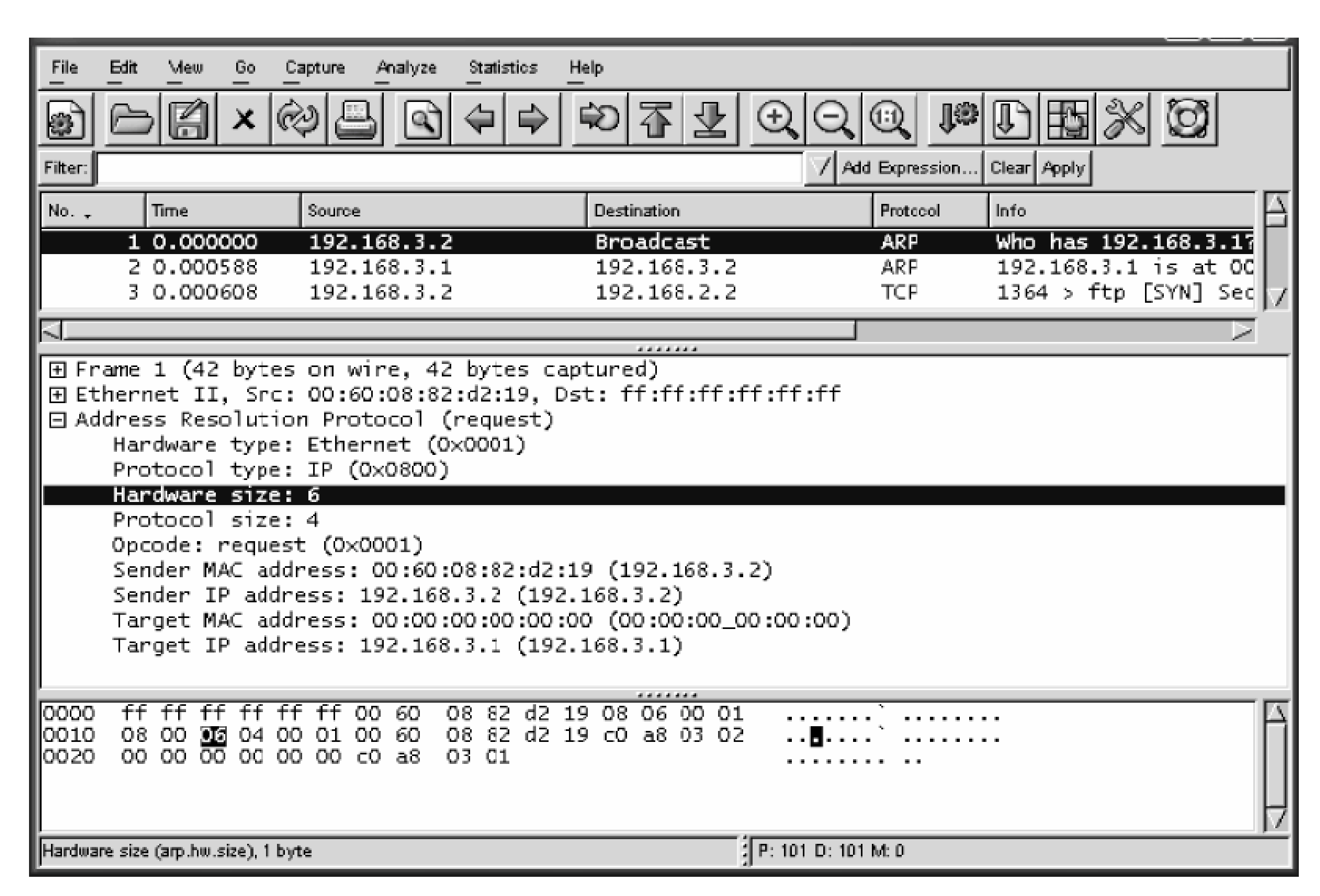



图 4-5

#### 4. Protocol Address Length

指定 Source 和 Destination Protocol Address 字段中找到的高层协议地址的长度(以字节为单位)。对于 IP 来说,这个值总是 4(32 位/8=4 字节),如图 4-6 所示。

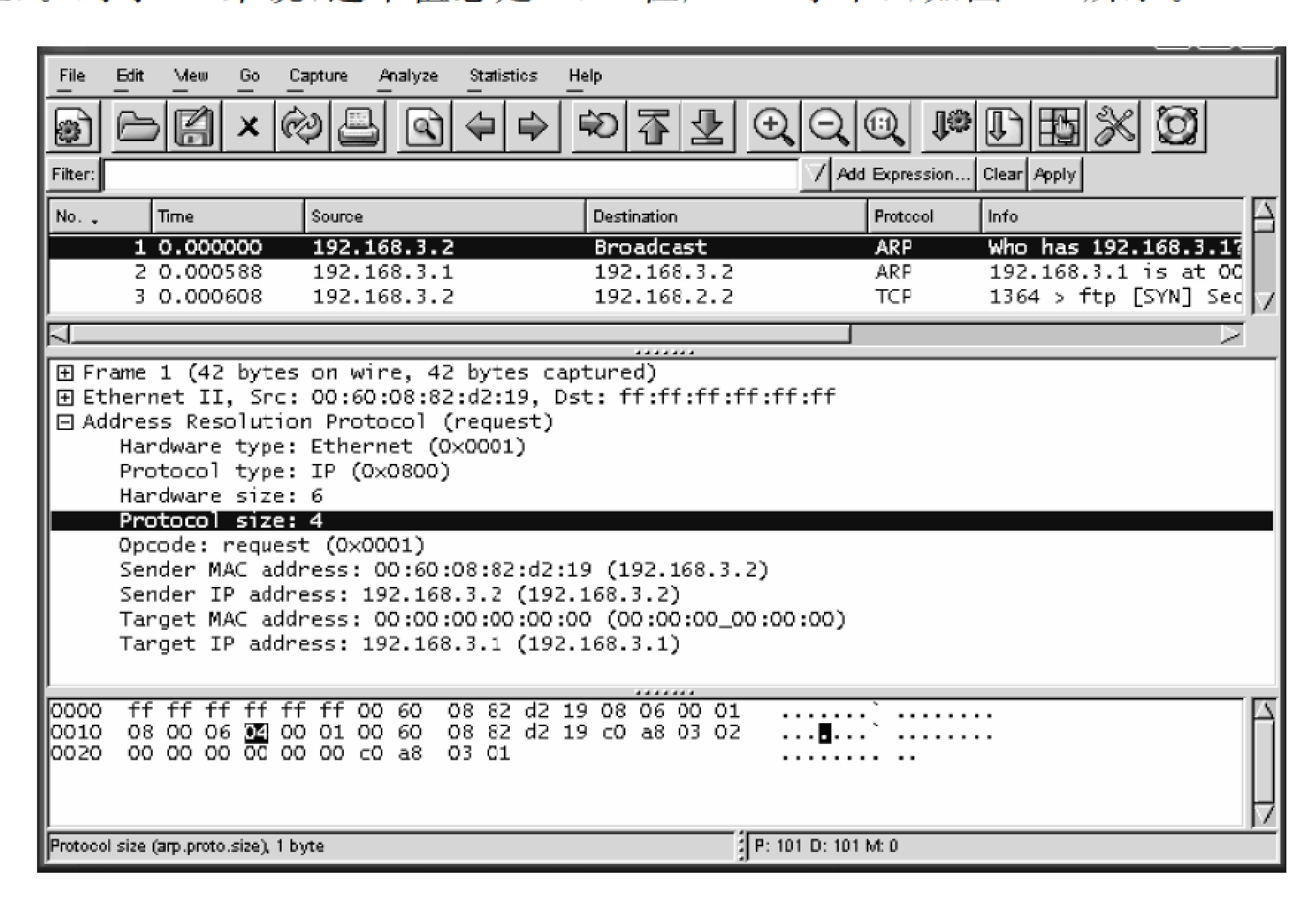


图　4-6

#### 5. Message Type

说明这个包的类型。

ARP 本身支持两种基本操作:请求与某一协议地址相关联的硬件地址,响应早些时候的请求。同样,RARP 也支持请求和响应操作的概念,这和 Inverse ARP 是一样的。

Message Type 类型和说明如表 4-2 所示。实际的 Message Type 如图 4-7 所示。

**表　4-2**

| 消 息 类 型 | 类 型 说 明 |
| --- | --- |
| 1 | ARP 请求 |
| 2 | ARP 响应 |
| 3 | RARP 请求 |
| 4 | RARP 响应 |
| 8 | Inverse ARP 请求 |
| 9 | Inverse ARP 响应 |

图 4-7 中用 Opcode 字段来表示 Message Type,这里是类型 0x0001,也就是类型 1,这是个 ARP 请求报文,这个请求报文中的目标 MAC 地址栏为 0。

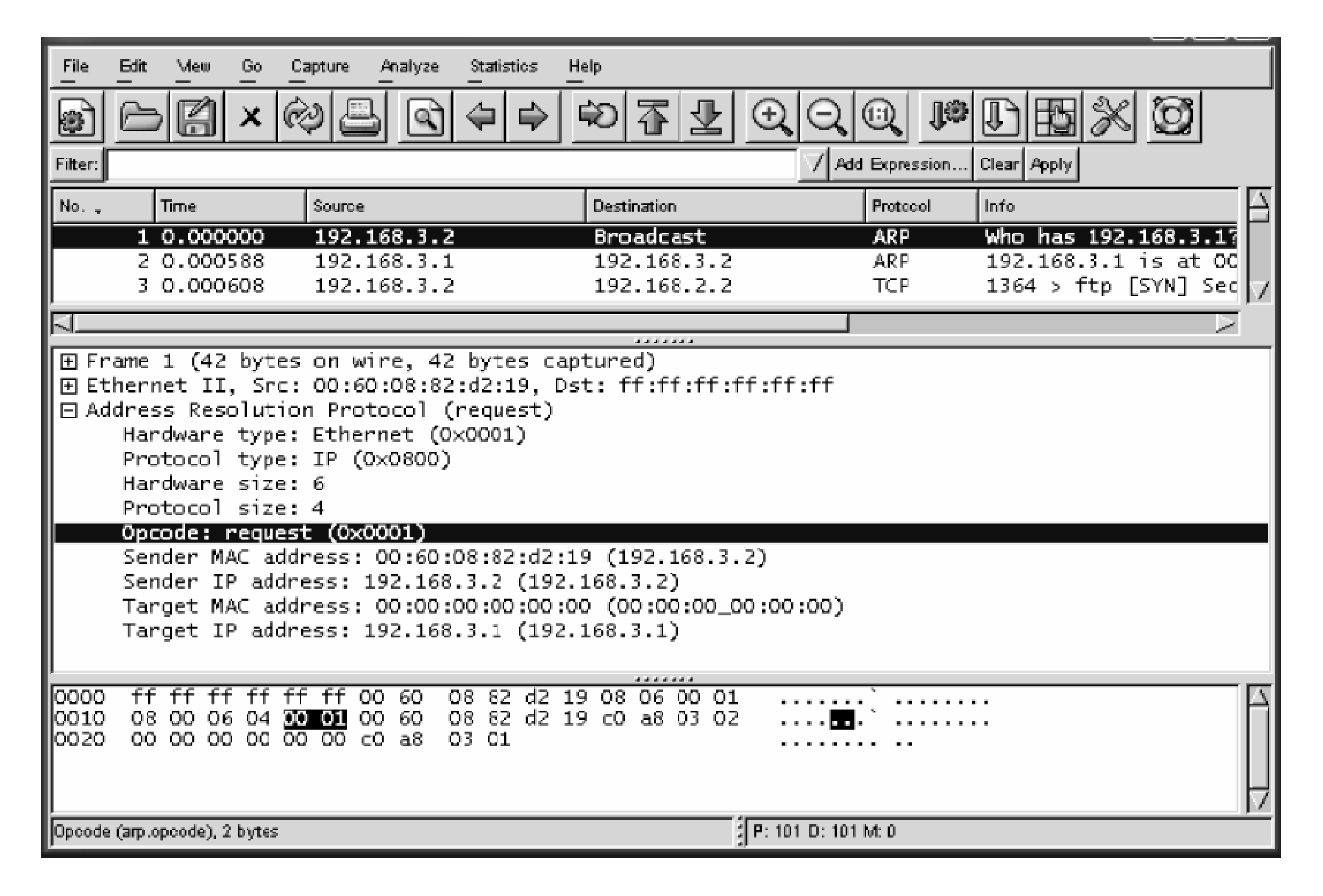



图 4-7

### 6. Source Hardware Address

标识发送该 ARP 包的主机的硬件地址。该包可以是请求也可以是响应。

每个 ARP 交换过程都包含两个独立的包：源包和对这个包的请求的响应。Source Hardware Address 字段说明了这个 ARP 包的发送者。如果这个包是一个请求，那么该字段包含的是发送该请求的设备的硬件地址。如果 ARP 包是一个响应，那么这个字段包含的是发送该响应的设备的硬件地址，如图 4-8 所示。

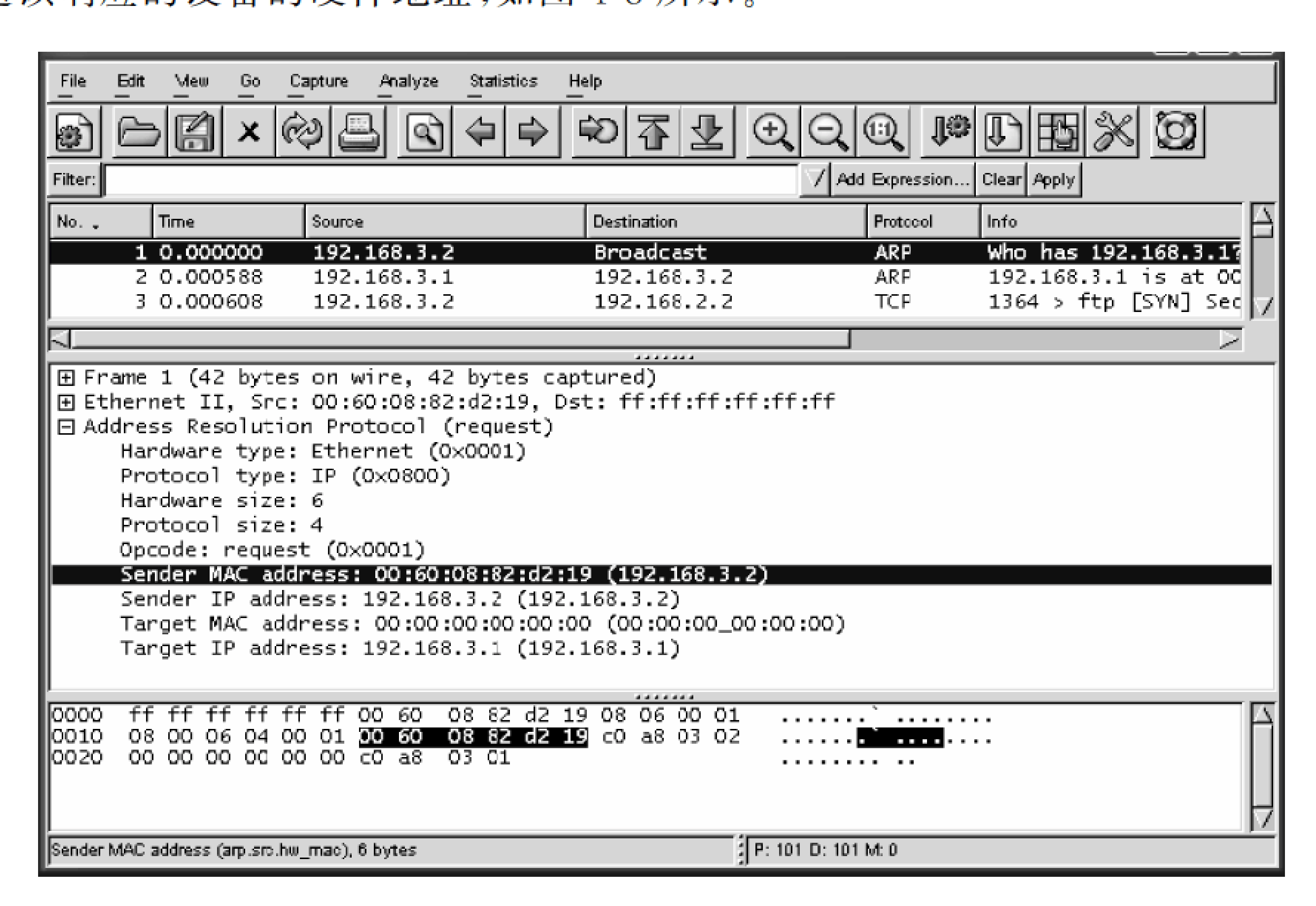



图 4-8

### 7. Source IP Address

标识发送该 ARP 包的主机的 IP 地址。

每个 ARP 交换都包含两个包：源请求和对这个包的请求的响应。Source IP Address 字段说明了这个 ARP 包的发送者的 IP 地址。如果这个包是一个请求，那么该字段包含的是发送该请求的设备的 IP 地址。如果 ARP 包是一个响应，那么这个字段包含的是发送该响应的设备的 IP 地址，如图 4-9 所示。

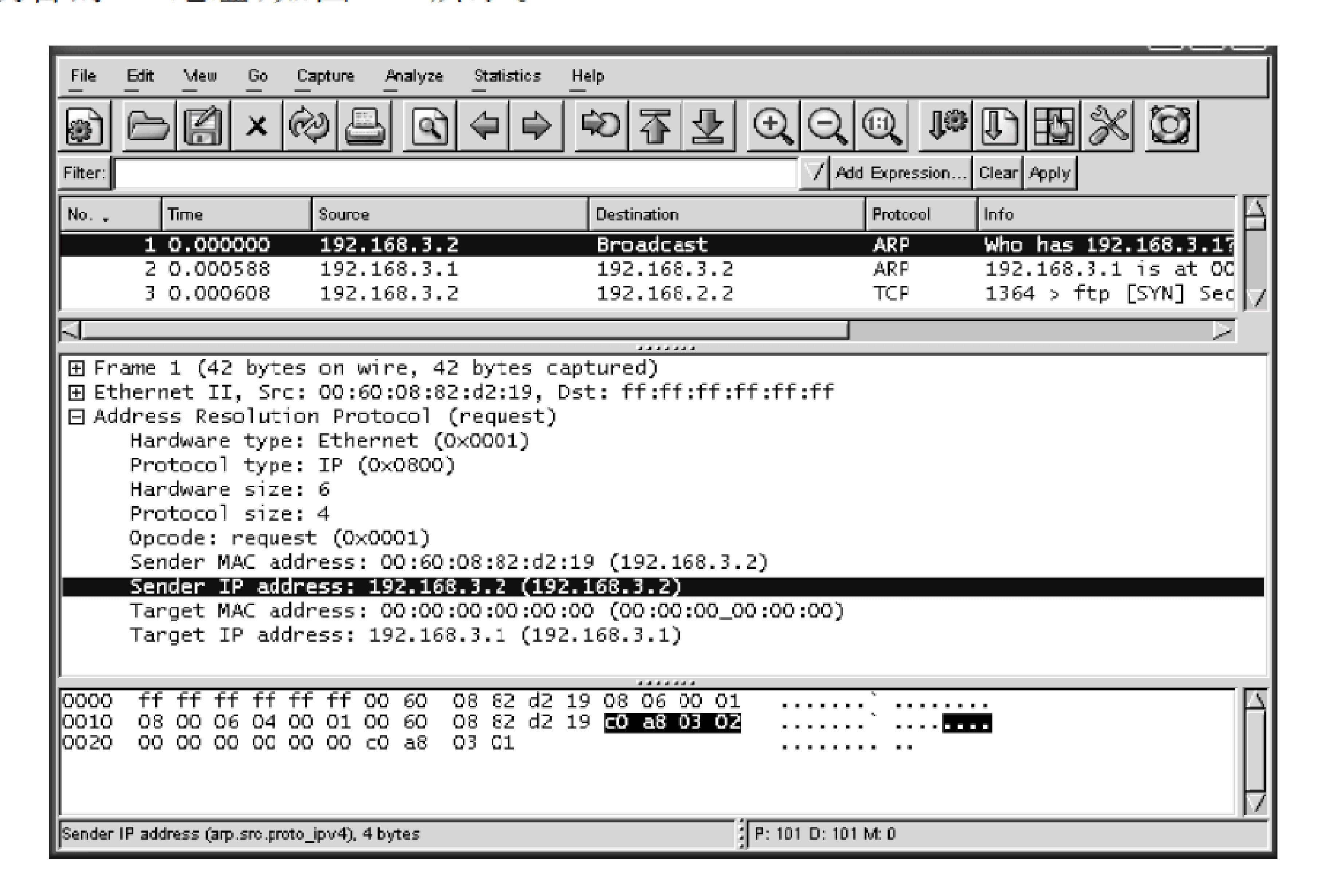



图　4-9

### 8. Destination Hardware Address

标识该 ARP 包的目的地的硬件地址，这个目的地可以是一个广播（请求中使用）或者某个特定地址（响应中使用）。

图 4-10 中 Destination Hardware Address 设为十六进制的 00：00：00：00：00：00，这说明发送者不知道目的地系统的硬件地址（说明 ARP 请求一个已知 IP 地址对应的硬件地址）。

### 9. Destination IP address

标识 ARP 报文的目的端的 IP 地址，如图 4-11 所示。

到此，讲了 ARP 包中的每个字段的意义及作用，接下来可以看到一个利用 ARP 来获得指定 IP 主机的 MAC 地址的完整过程。

图 4-12 显示主机 192.168.3.2 发送一个 ARP 请求包 request，这是一个广播，查找网络中 192.168.3.1 的 MAC 地址，可以看到，它是通过一个广播地址完成的。

图 4-13 显示了一个 ARP 回应包 reply，它发送的目标地址是 192.168.3.2，源地址是 192.168.3.1，可以看到 ARP 包中包含了 192.168.3.1 的 MAC 地址。这样，192.168.3.2 就得到了 192.168.3.1 的硬件 MAC 地址。

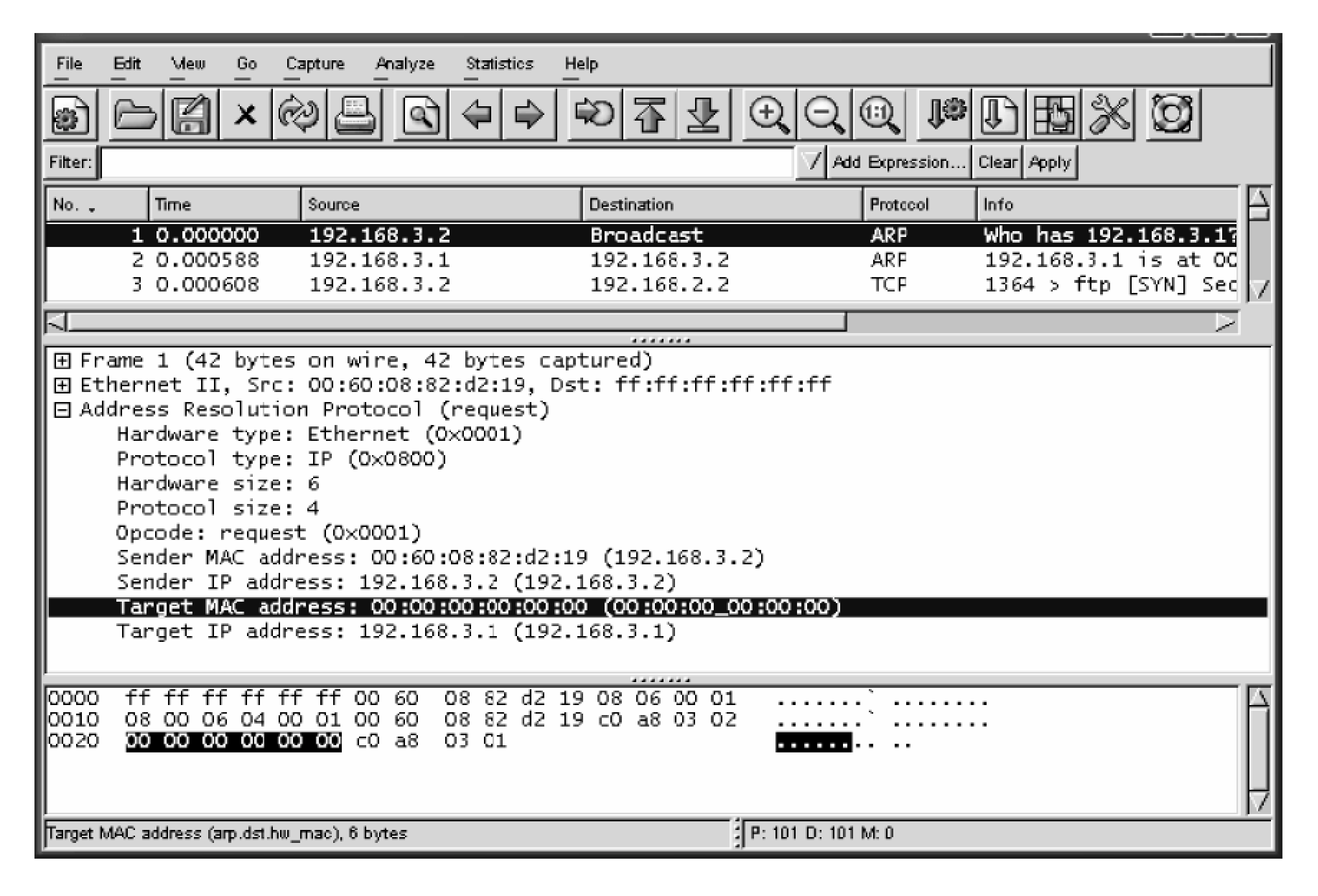



图 4-10

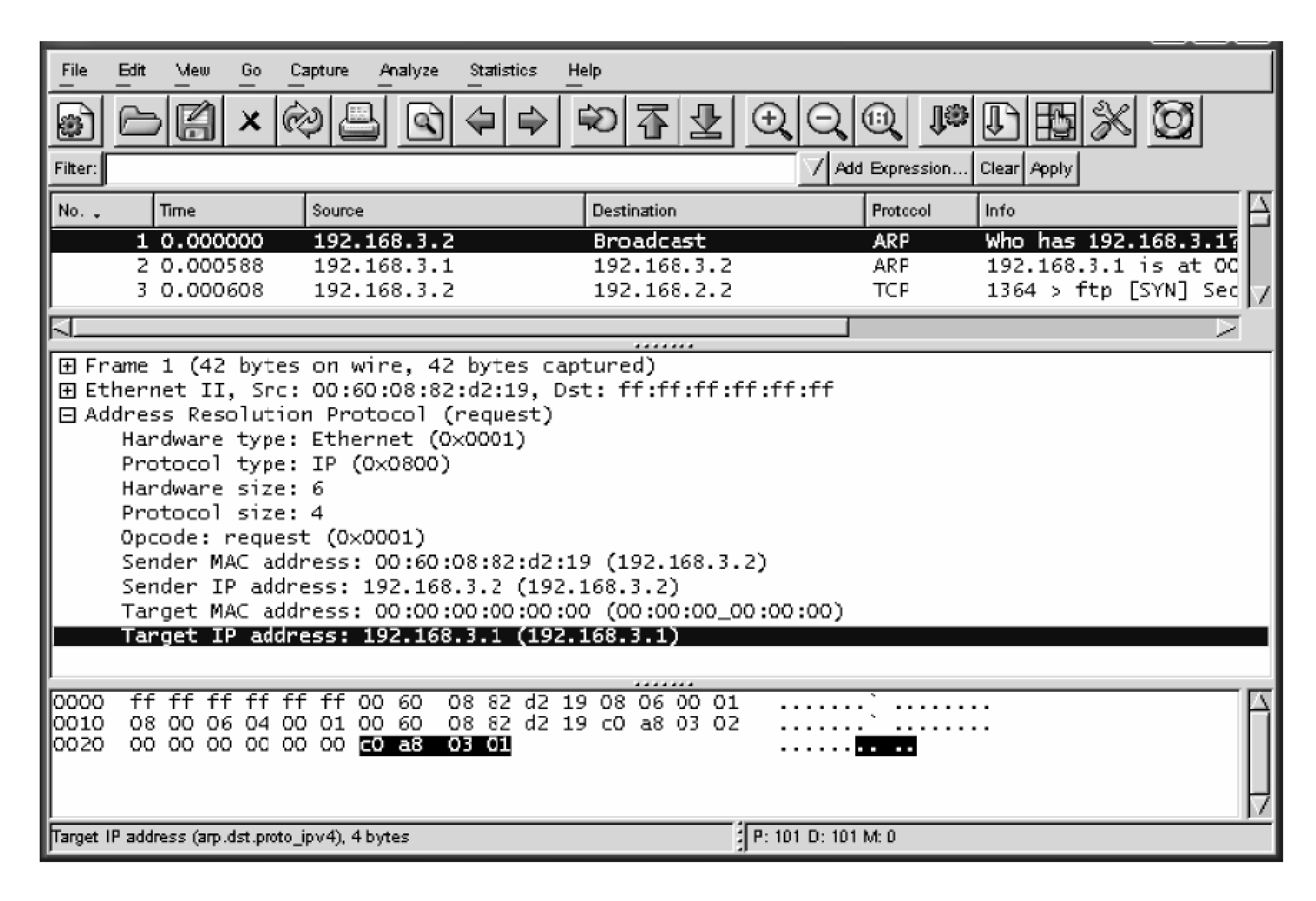



图 4-11

上面讲的是已知 IP 地址利用 ARP 来获得该 IP 地址的 MAC 地址，接下来看看 ARP 的另一个用途——重复地址检查，这种 ARP 叫 Gratuitous ARP(GARP)。当主机第一次获得 IP 地址时，会使用自己的 IP 地址作为目标地址发送 ARP 请求，以确定自己使用的 IP 地址是否有人在用，如果收到 ARP 响应则表明这个 IP 地址已经有人使用，这种 ARP 请求称为无故 ARP、GARP，如图 4-14 所示。

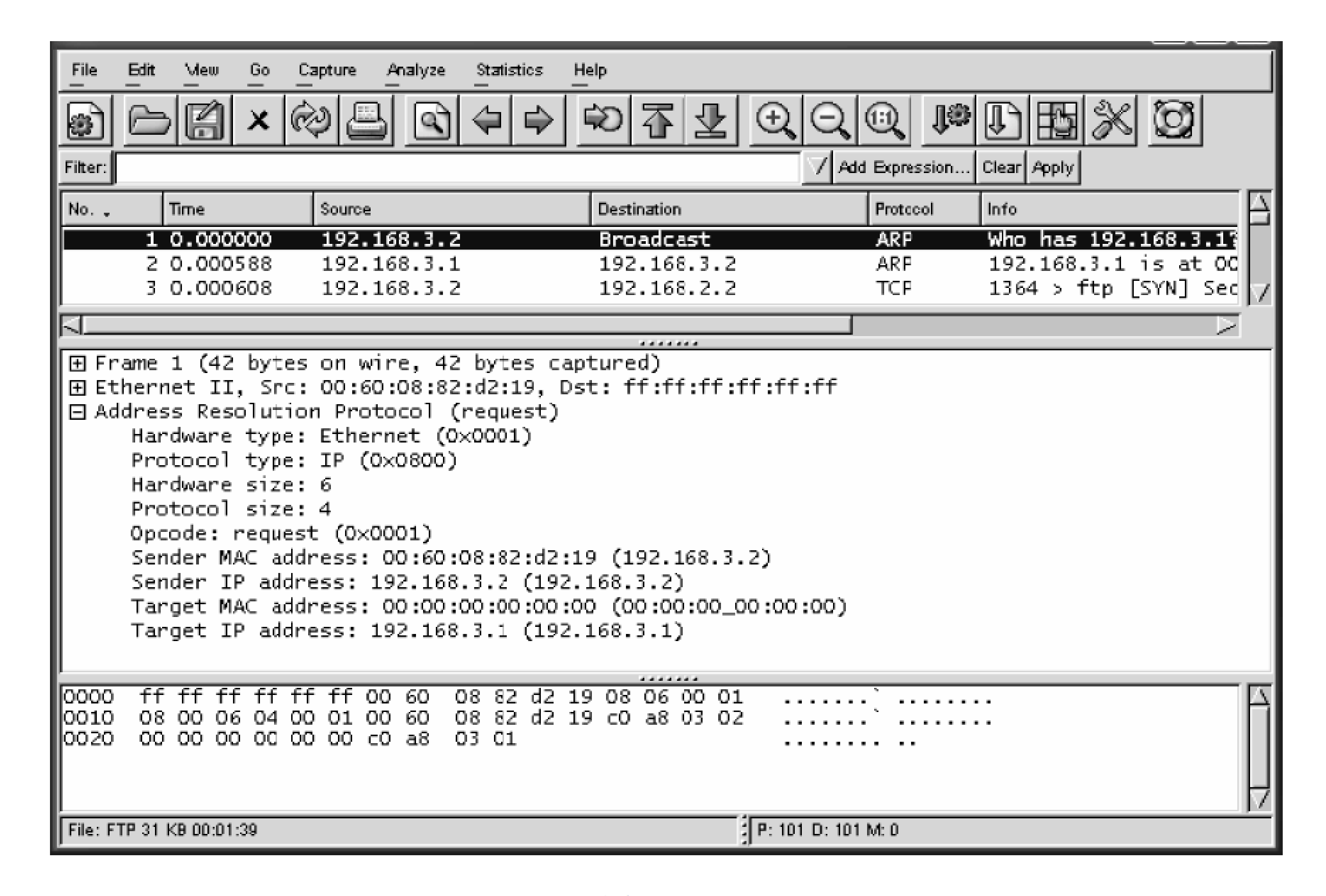



图　4-12

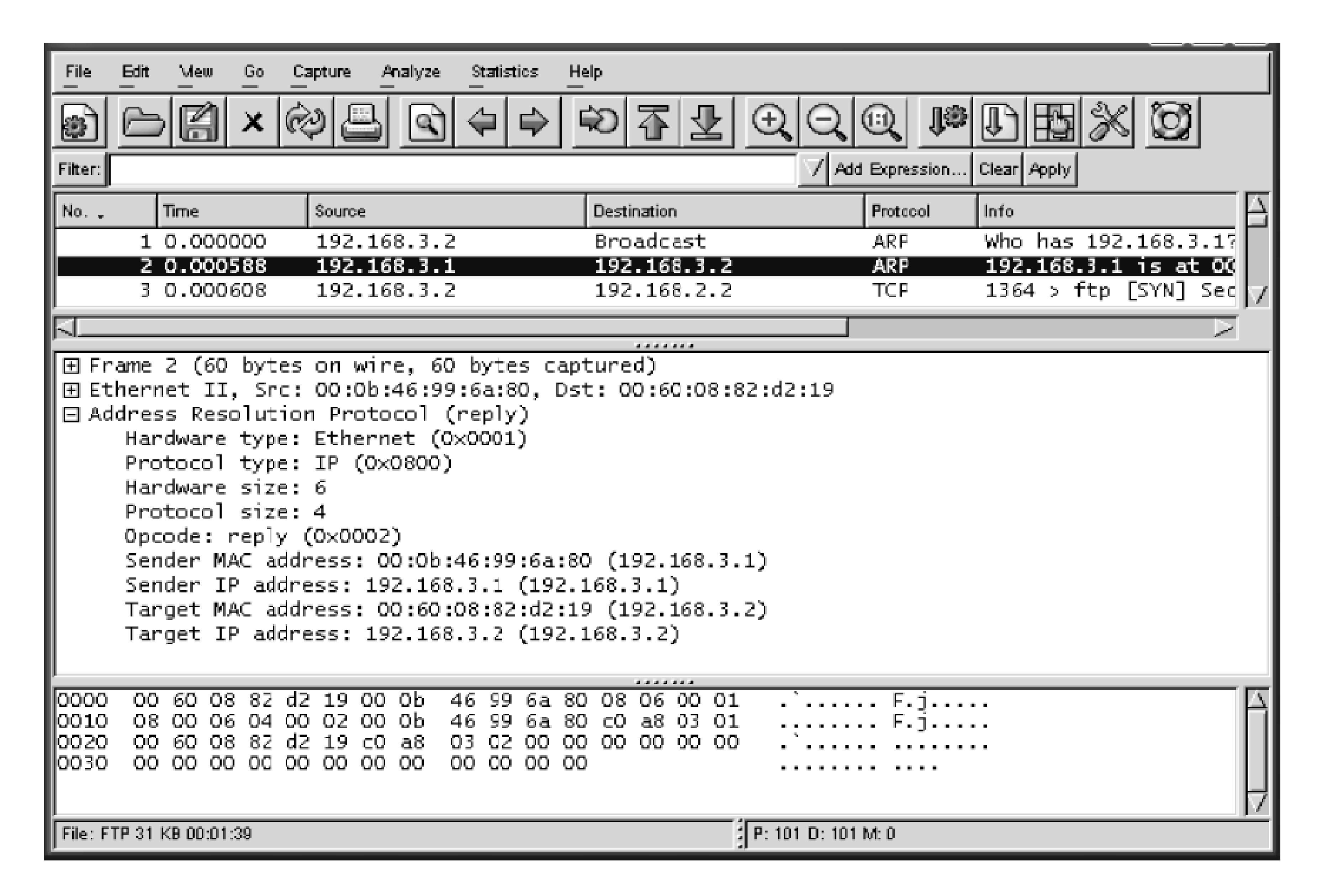



图　4-13

图 4-14 中，ARP 是个 ARP 请求，这个 ARP 包的源硬件地址（MAC 地址）是发送主机的 MAC 地址，目标硬件地址（MAC 地址）使用的是广播地址（FF-FF-FF-FF-FF-FF）；而源和目标协议地址（IP 地址）都是此发送主机的 IP 地址。表明主机想检测 IP 地址 10.61.19.2 是否有人在用。

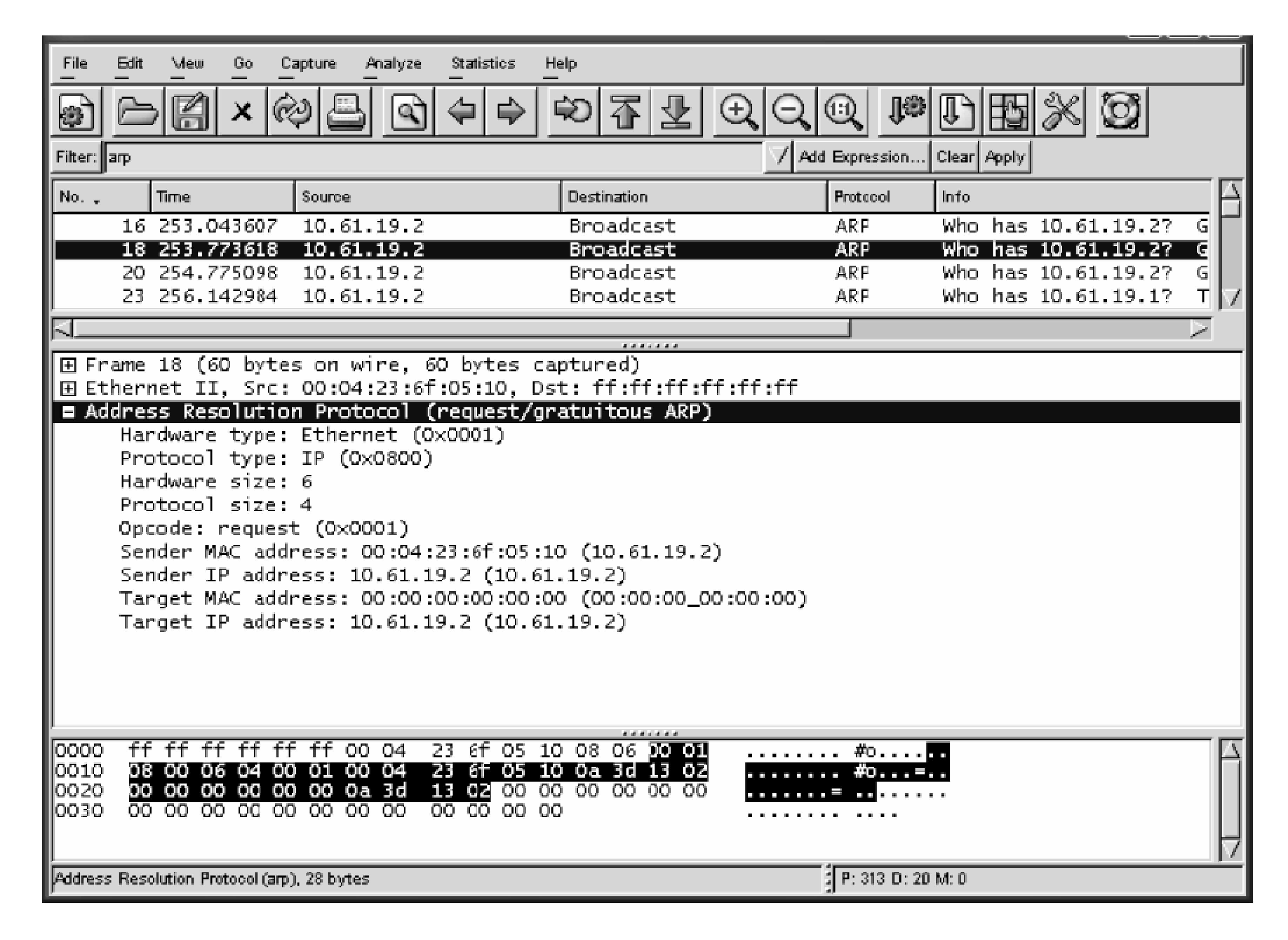

File Edit View Go Capture Analyze Statistics Help
Filter: arp
Add Expression... Clear Apply
No. Time Source Destination Protocol Info
16 253.043607 10.61.19.2 Broadcast ARP Who has 10.61.19.2? G
18 253.773618 10.61.19.2 Broadcast ARP Who has 10.61.19.2? G
20 254.775098 10.61.19.2 Broadcast ARP Who has 10.61.19.2? G
23 256.142984 10.61.19.2 Broadcast ARP Who has 10.61.19.1? T
Frame 18 (60 bytes on wire, 60 bytes captured)
Ethernet II, Src: 00:04:23:6f:05:10, Dst: ff:ff:ff:ff:ff:ff
Address Resolution Protocol (request/gratuitous ARP)
Hardware type: Ethernet (0x0001)
Protocol type: IP (0x0800)
Hardware size: 6
Protocol size: 4
Opcode: request (0x0001)
Sender MAC address: 00:04:23:6f:05:10 (10.61.19.2)
Sender IP address: 10.61.19.2 (10.61.19.2)
Target MAC address: 00:00:00:00:00:00 (00:00:00_00:00:00)
Target IP address: 10.61.19.2 (10.61.19.2)
0000 ff ff ff ff ff ff 00 04 23 6f 05 10 08 06 00 01
0010 08 00 06 04 00 01 00 04 23 6f 05 10 0a 3d 13 02
0020 00 00 00 00 00 00 0a 3d 13 02 00 00 00 00 00 00
0030 00 00 00 00 00 00 00 00 00 00 00 00
Address Resolution Protocol (arp), 28 bytes
P: 313 D: 20 M: 0

图 4-14

CHAPTER 5

# 第5章 ICMP 协议

ICMP 不是传输协议，也不能用来发送应用程序数据，ICMP 是消息控制协议，它的作用是发送消息把网络事件和变化告诉设备。

由于 IP 协议是一种不可靠的协议，无法进行差错控制。IP 负责把数据从一个主机传输到另一个主机上，有时从一个网络传输到另一个网络上。在传输过程中，IP 数据都可能因为某种原因不能发送，这样根据失败的性质决定是否让发送端知道失败原因。

比如，无效的校验和这样的暂时性错误是被忽略掉的，因为下一个包不太可能会出现同样的问题，最终所用的传输或应用程序会检测到错误，这个失败并不表示这个网络出现了普遍错误；相反，半永久性错误（诸如目的 IP 地址不可达）需要立刻报告给发送者，因为这个失败表明了网络自身出现了重要问题。

ICMP 是在检测到半永久性错误时发送失败消息的协议。这些错误包含：目的地不可达、IP Time-to-live 值为 0 等。除了发送错误消息，ICMP 可以用来交换关于网络的一般信息，或者用来探测网络的一定特征。例如，常用的 ping 程序就用 ICMP 消息来检测两个设备之间的基本连接。

ICMP 是一个预定义好的消息的集合，每一种消息都提供专用的功能，每一种类型的 ICMP 消息称为“主要类型”，“主要类型”还拥有自己的“子类型编码”。当系统需要发送 ICMP 消息时，它就从消息库里找出一条消息，把这个消息的代码放入 ICMP 数据包文中，然后通过 IP 把 ICMP 消息发送到源端。

虽然 ICMP 可以用来报告 IP 的失败，但应该注意 ICMP 不会使 IP 协议变得可靠。IP 仍然允许丢失包、发送重复包、非顺序包或其他的任何事情。主机可能会因为很多原因忽略这些消息，而不给发送者返回 ICMP 消息，因此没有 ICMP 消息也不意味着网络工作正常。

通常人们使用的 ping 命令，就是使用 ICMP 的 Echo Request 和 Echo Reply 消息，此外 ICMP 还提供了 Error 错误消息。

ICMP Echo Request(Reply)报文结构如图 5-1 所示，它包括类型、代码、校验和、标识符和序列号，后面就是 ICMP 的数据。其中类型占 8 位，代码占 8 位，校验和占 16 位，这三个字段共 32 位，在任何 ICMP 报文中是不变的，后面的字段结构根据不同的 ICMP 类型和代码有所不同，如图 5-2 所示，这是 ICMP 通用格式。

| 类型 | 代码 | 校验和 |
|---|---|---|
| 标识符 | | 序列号 |
| 数据 | | |

图 5-1　ICMP Echo Request(Reply)报文结构

| 类型 | 代码 | 校验和 |
|---|---|---|
| 不同的类型和代码，内容不同 | | |

图 5-2　ICMP 通用格式

## 5.1　Echo Request 和 Echo Reply 查询消息

ICMP 提供了两个查询消息 Echo Request 和 Echo Reply，这两个查询消息放在一起可以测试网络上的某远程系统是否在工作。ICMP Echo Request 查询消息是用户发送到目的主机的探测包，并由 ICMP Echo Reply 查询消息来响应。

比如测试网络中主机的基本连接时，使用 ping 程序来测试联通性，ping 的工作原理就是通过往目的系统发送一个或更多个 ICMP Echo Request 消息，然后计算从探测开始到收到 ICMP Echo Reply 消息这个过程所花费的时间。

图 5-3 和图 5-4 显示本地网络上两个主机之间一个简单的 ping。例子中 192.168.2.3 在 ping 主机 192.168.2.2，它们之间通过发送 ICMP Echo request 和 ICMP Echo Reply 来完成，具体的详细数据包的分析见后面相关 ICMP 包的分析。

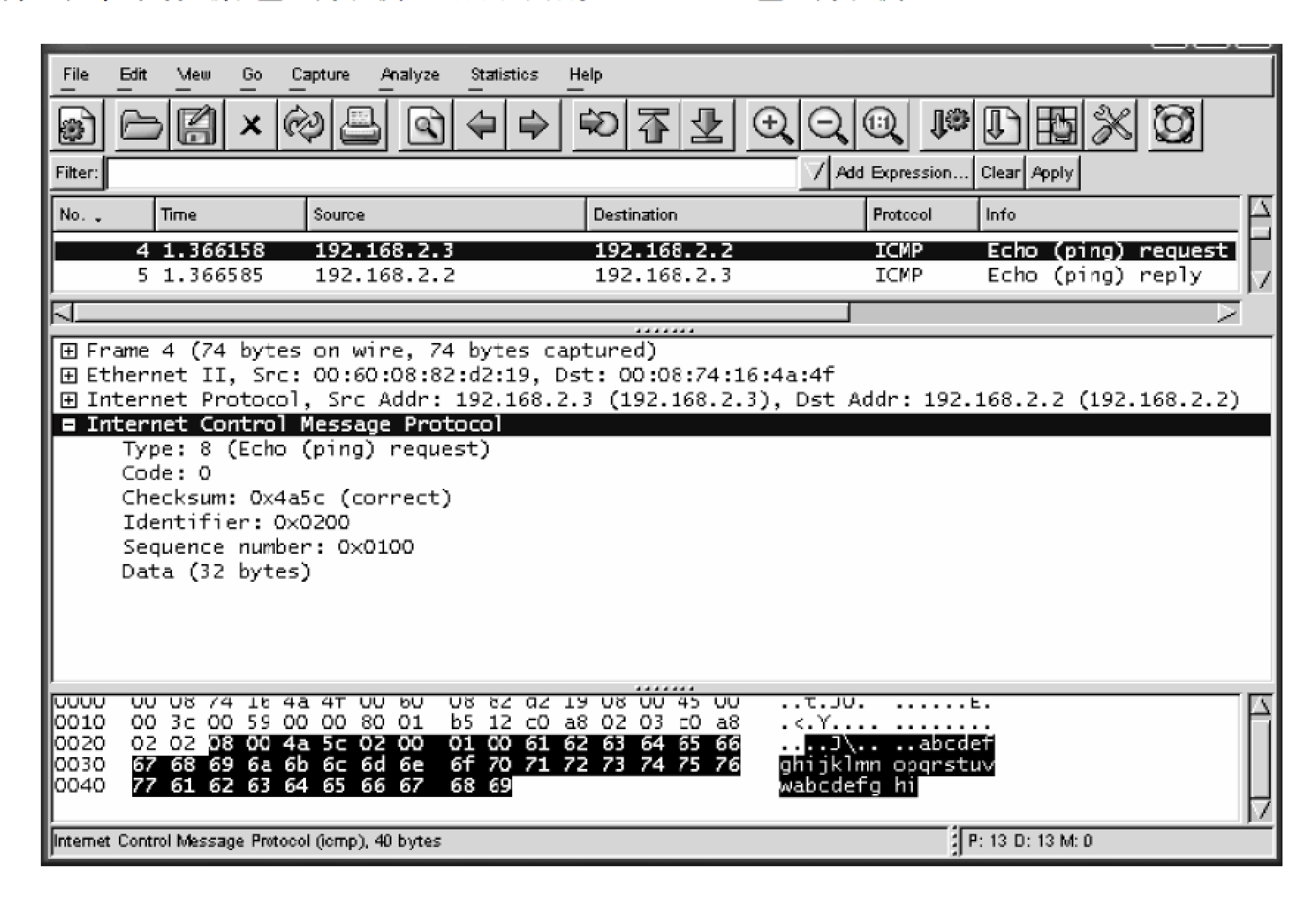



图　5-3

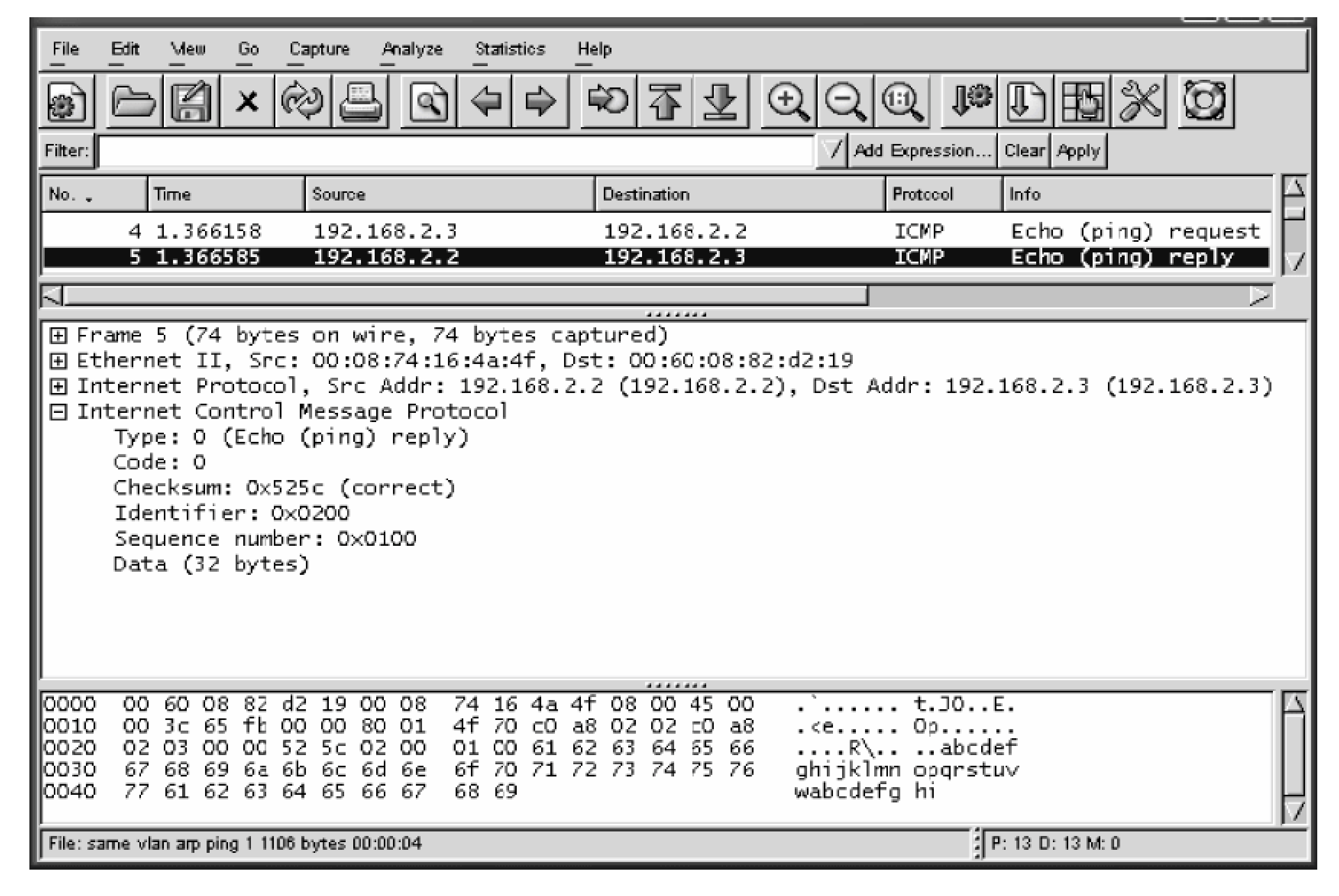



图　5-4

## 5.2　ICMP 消息类型

ping 程序只是 ICMP 应用的其中一种形式，ICMP 还存在其他应用中。下面来讲讲 ICMP 的其他一些类型。

每个 ICMP 消息都指定了一个唯一的“消息类型”(Message Type)也就是前面说的主要类型，它只是一个简单的数字代码来代表每种预定好的信息，在设备需要往另外一个设备上发送信息时，可以选择几种预先定义好的消息类型。比如上面的 ping 程序，它用到的 ICMP Message Type 值 Echo Reply 是 0，Echo Request 是 8，从图 5-3 和 5-4 中可以看到。表 5-1 是 IPv4 常用的 Message Type。

**表　5-1**

| TYPE | 消 息 描 述 | 消 息 种 类 | 定 义 源 |
|---|---|---|---|
| 0 | Echo reply | Query (reply) | RFC 192 |
| 3 | Destination Unreachable | Error | RFC 1122 |
| 4 | Source Quench | Error | RFC 792 |
| 5 | Redirect | Error | RFC 792 |
| 8 | Echo Request | Query (request) | RFC 792 |
| 9 | Router Advertisement | Query (reply) | RFC 1256 |
| 10 | Router Solicitation | Query (request) | RFC 1256 |
| 11 | Time Exceeded | Error | RFC 1122 |
| 12 | Parameter Problem | Error | RFC 792 |
| 13 | Timestamp Request | Query (request) | RFC 792 |
| 14 | Timestamp Reply | Query (reply) | RFC 792 |
| 17 | Address Mask Request | Query (request) | RFC 950 |
| 18 | Address Mask Reply | Query (reply) | RFC 950 |

从表 5-1 可以看到多种消息类型(Message Type),不同的 Type 对应了不同的应用,除了查询消息外,ICMP 还提供了 Error 错误消息。

## 5.3 ICMP 各字段分析

在分析 ICMP 各字段意义前,在第 1 章中说过为 ICMP 分层是个棘手的问题,从图 5-5 中看到,实际上 ICMP 是封装在 IP 数据包中,从图中可以看到 IP 数据包中有个 Protocol Type:ICMP 字段,表明这个 IP 数据包的上一层协议是 ICMP。

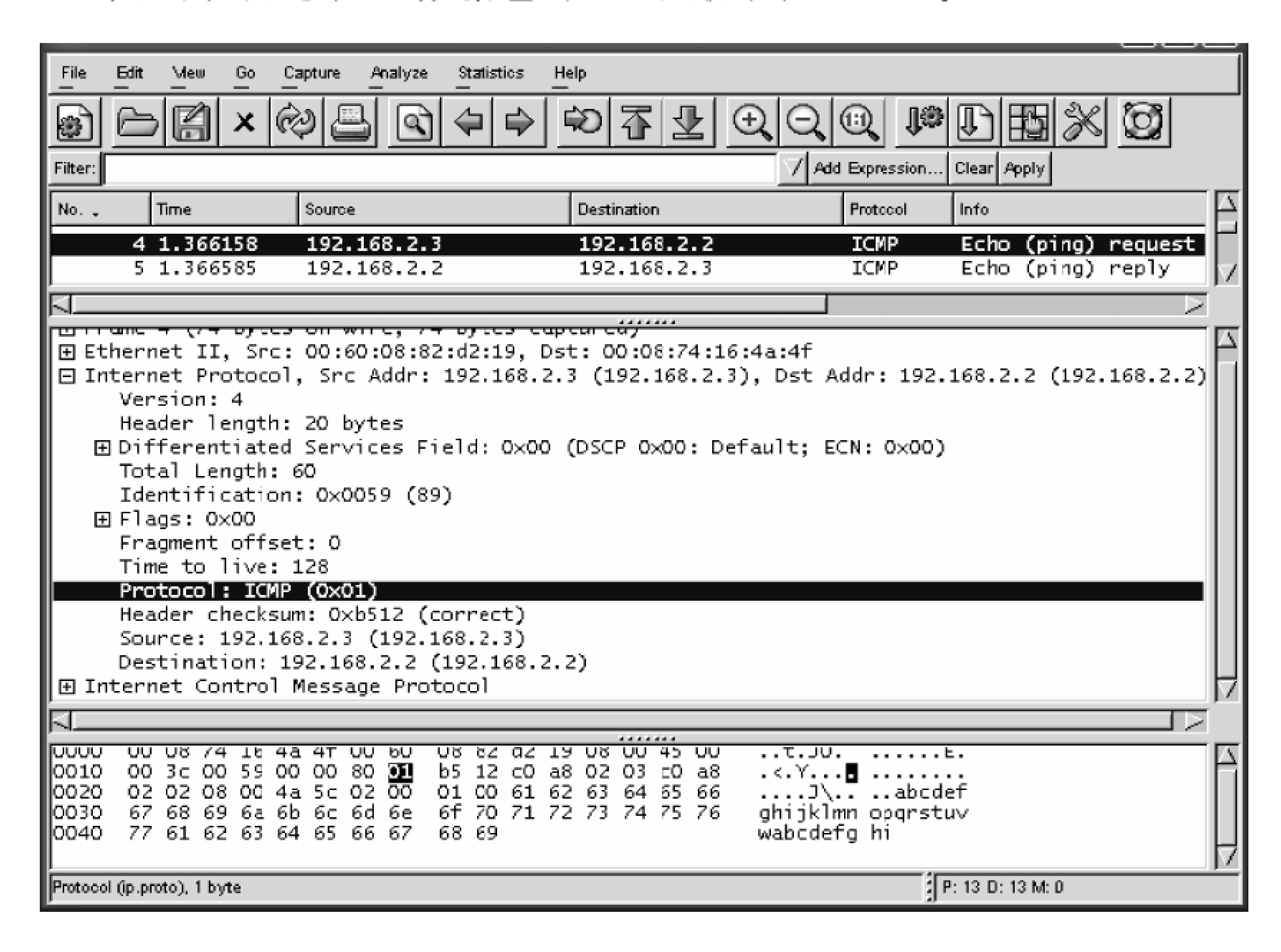



图 5-5

各种 ICMP 报文的前 32 位都一样,接下来对 ICMP 数据包中的每个字段进行介绍。

**1. Message Type 字段:8 位**

从表 5-1 可知,因为图 5-6 中是个 Echo Request,所以 Message Type 值是 8。

**2. Message Code 字段:8 位**

前面讲过 ICMP 除了 Echo request 和 Echo Reply 消息外,ICMP 还提供了 Error 错误消息,这个字段就是用来说明特定 ICMP 错误消息子类,也就是前面说的“子类型编码(Minor Codes)”。

因为图 5-7 并没有出错,所以图 5-7 中 Code=0。在表 5-2 列出了常见的 Code 含义。

举个出错的例子:如果 Code=3,当和 type=3 (Destination Unreachable)错误消息一起使用时,这个 3 表示子类型为 Port Unreachable。

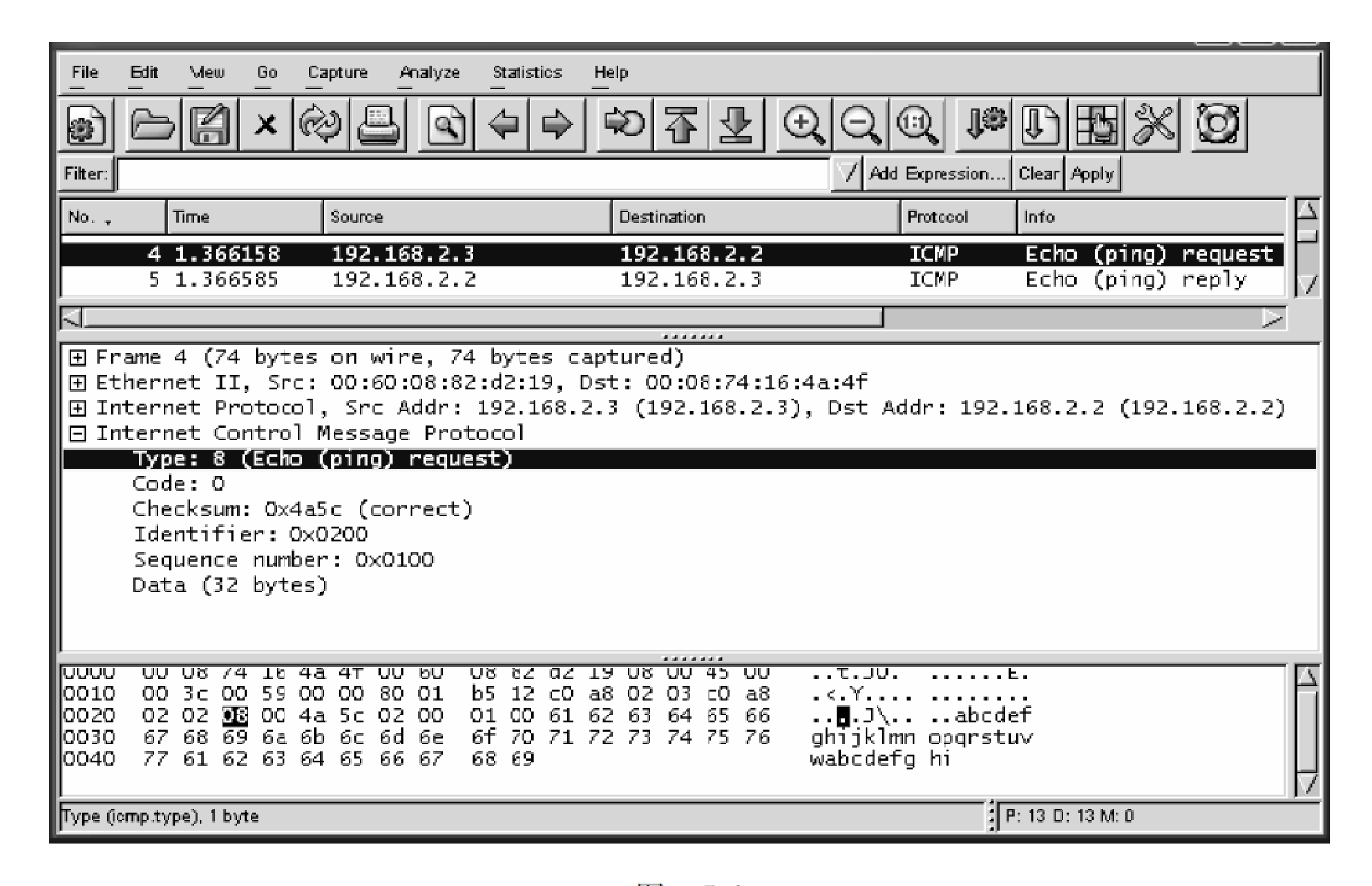



图　5-6

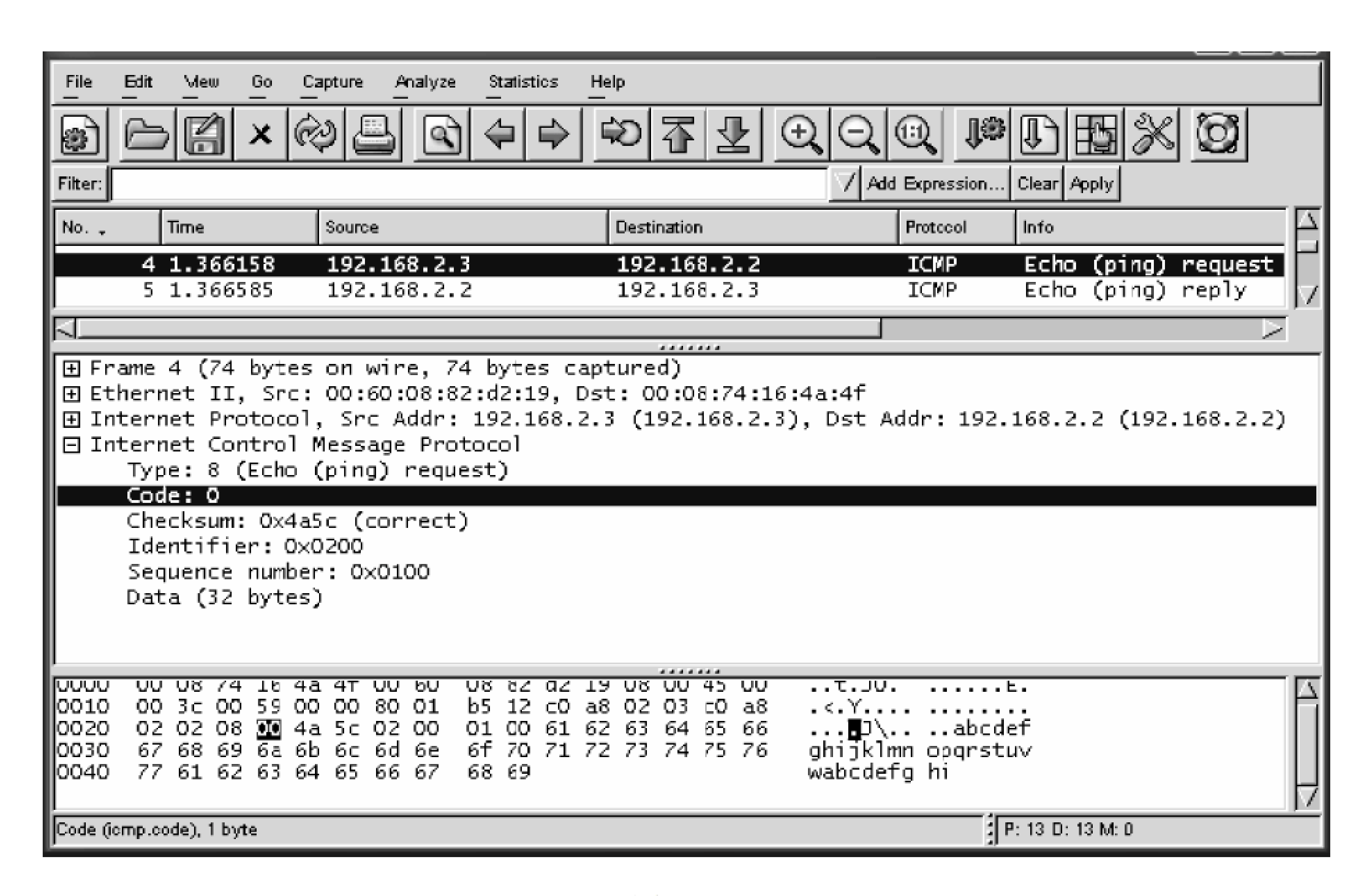



图　5-7

具体可查表 5-2 对应子类，ICMP 实质上包含一个预先定义好的消息字典，允许通过数字标识符来交换数据。Type 字段是用来说明信息的主要类，而 Code 字段是用来说明小类别。但是一些信息 Type 不具有任何子类型 Code。

表 5-2

| Type | Code | 描　　述 | 消 息 种 类 |
|---|---|---|---|
| 0 | 0 | 回应的应答 | Query(Reply) |
| 3 | | 目标不可达 | Error |
| | 0 | 网络不可达 | Error |
| | 1 | 主机不可达 | Error |
| | 2 | 协议不可达 | Error |
| | 3 | 端口不可达 | Error |
| | 4 | 需要分片但设置了不可分片位 | Error |
| | 5 | 源站选路失败 | Error |
| | 6 | 目的网络不认识 | Error |
| | 7 | 目的主机不认识 | Error |
| | 8 | 源主机被隔离(作废不用) | Error |
| | 9 | 目的网络被强制禁止 | Error |
| | 10 | 目的主机被强制禁止 | Error |
| | 11 | 由于服务类型 TOS,网络不可达 | Error |
| | 12 | 由于服务类型 TOS,主机不可达 | Error |
| | 13 | 由于过滤,通信被强制禁止 | Error |
| | 14 | 主机越权 | Error |
| | 15 | 优先权中止生效 | Error |
| 4 | 0 | 源端被关闭(基本流控制) | Error |
| 5 | | 重定向 | Error |
| | 0 | 对网络重定向 | Error |
| | 1 | 对主机重定向 | Error |
| | 2 | 对服务类型和网络重定向 | Error |
| | 3 | 对服务类型和主机重定向 | Error |
| 8 | 0 | 回应请求 | Query(Reply) |
| 9 | 0 | 路由器通告 | Query(Reply) |
| 10 | 0 | 路由器请求 | Query(Reply) |
| 11 | | 超时 | Error |
| | 0 | 传输期间生存时间为 0 | Error |
| | 1 | 在数据包组装期间生存时间为 0 | Error |
| 12 | | 参数问题 | Error |
| | 0 | 坏的 IP 头部(包括各种差错) | Error |
| | 1 | 缺少必要的选项 | Error |
| 13 | 0 | 时间戳请求 | Query(Reply) |
| 14 | 0 | 时间戳应答 | Query(Reply) |
| 17 | 0 | 地址掩码请求 | Query(Reply) |
| 18 | 0 | 地址掩码应答 | Query(Reply) |

### 3. Checksum 字段：16 位

Checksum 字段包括数据在内的整个 ICMP 数据包的校验和,其计算方法和 IP 头部校验和的计算方法是一样的,如图 5-8 所示。

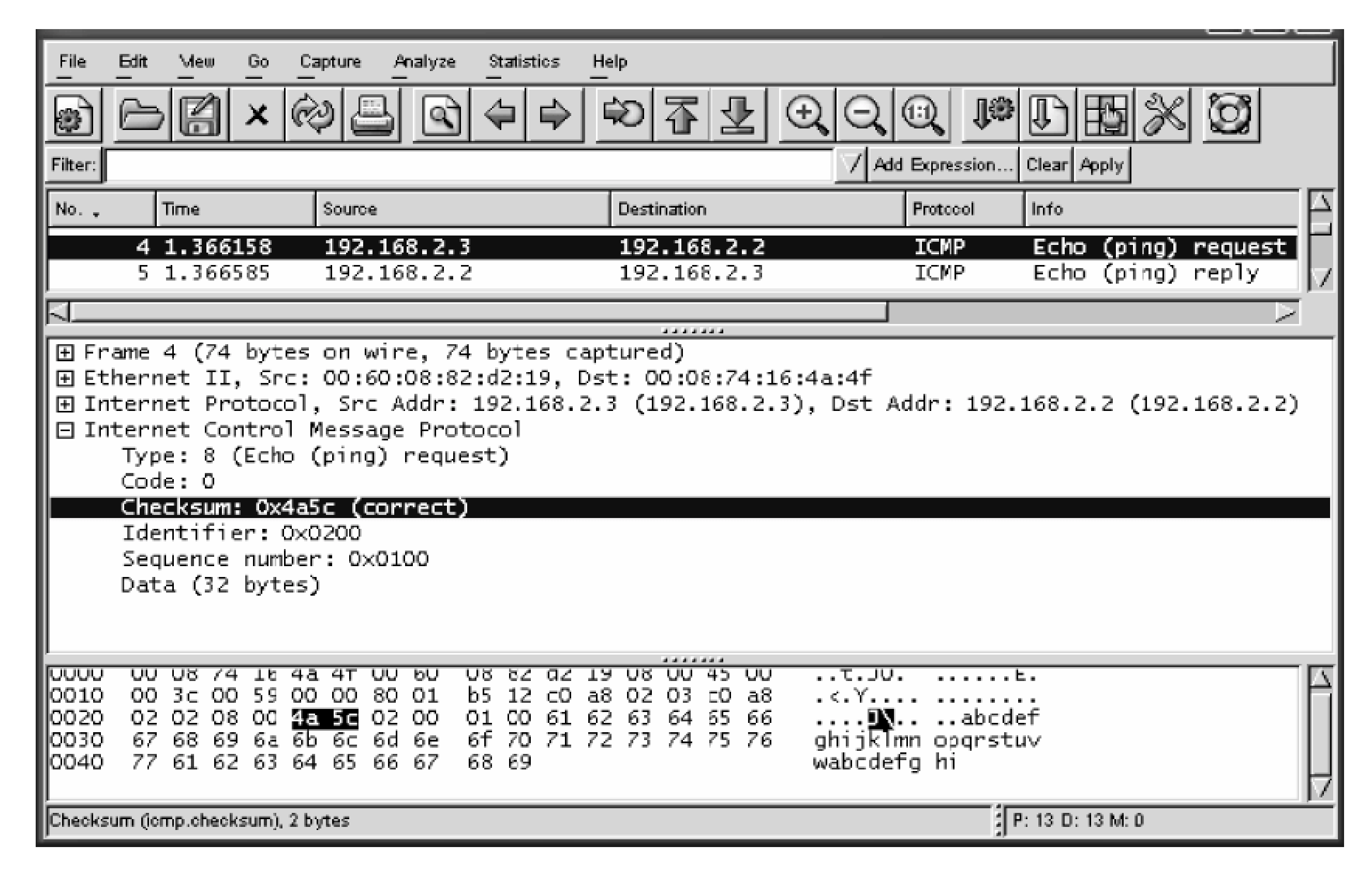



图　5-8

前面 3 个字段是所有 ICMP 报文共有的，下面的字段对于本例是针对 ping 应用来讲。

（1）Identifier 字段

Identifier 字段标识本 ICMP 进程，如图 5-9 所示，这个字段长度 16 位。

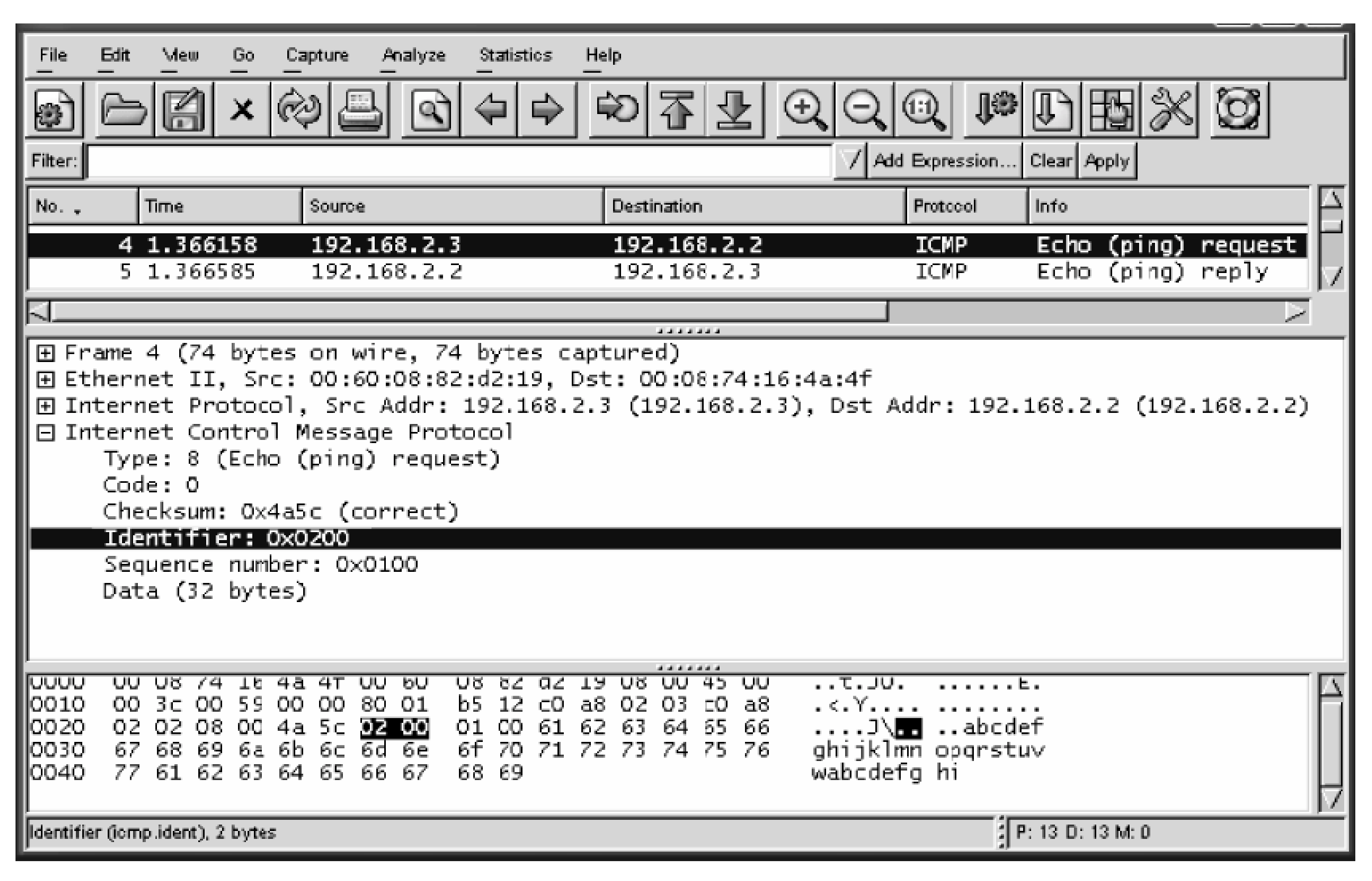



图　5-9

（2）Sequence 字段

用于判断回应应答数据包，如图 5-10 所示。

（3）Data 字段

ICMP Data 字段最大允许的长度为 1472。平时在用 ping 命令时，Windows 下默认发送的 Data 是 32 字节，如图 5-11 所示。如果发送一个 400 字节大小的 ping 数据，这时的 Data 部分就是 400 字节了。

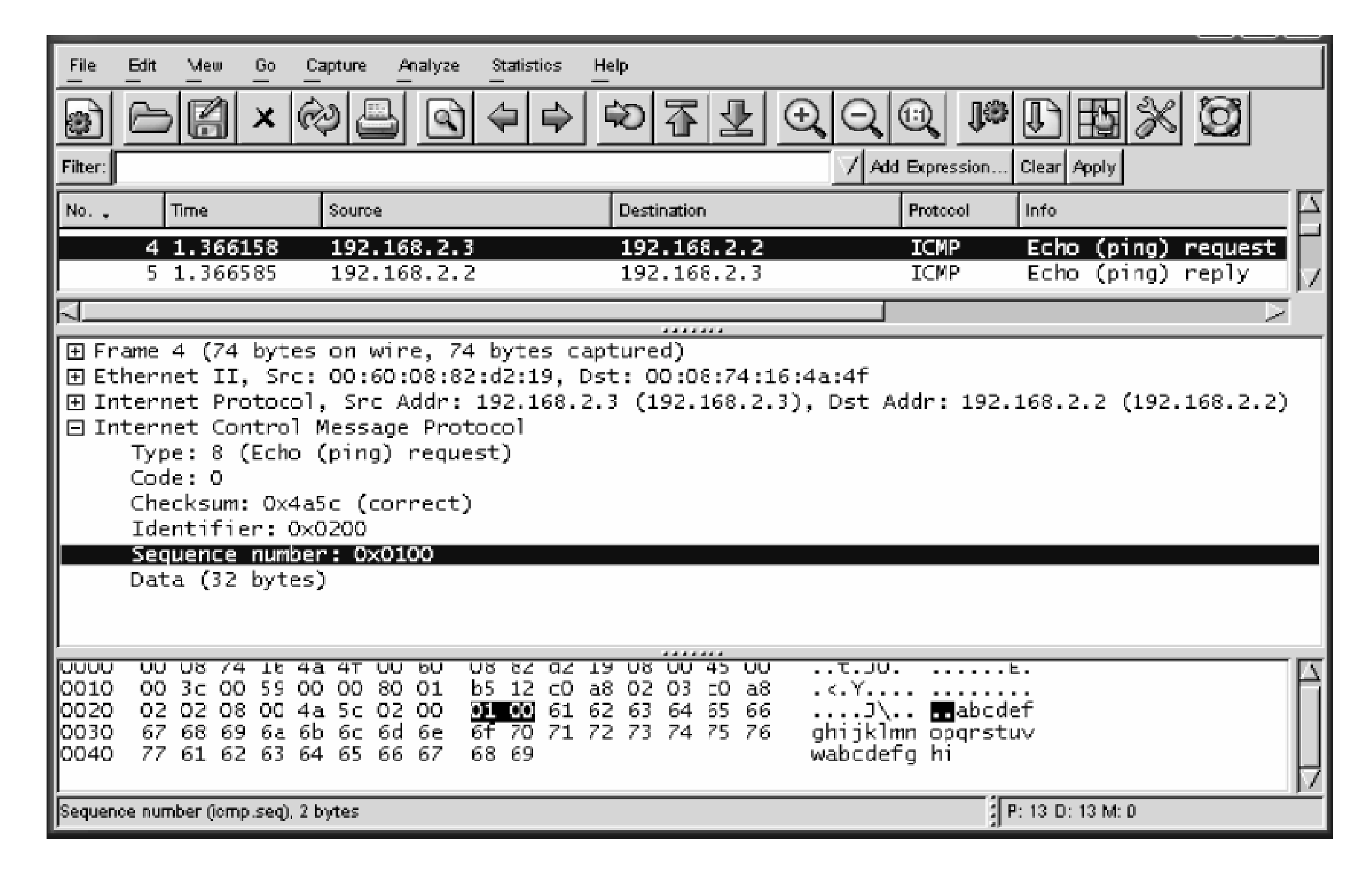

File Edit View Go Capture Analyze Statistics Help
Filter: Add Expression... Clear Apply
No. Time Source Destination Protocol Info
4 1.366158 192.168.2.3 192.168.2.2 ICMP Echo (ping) request
5 1.366585 192.168.2.2 192.168.2.3 ICMP Echo (ping) reply
Frame 4 (74 bytes on wire, 74 bytes captured)
Ethernet II, Src: 00:60:08:82:d2:19, Dst: 00:08:74:16:4a:4f
Internet Protocol, Src Addr: 192.168.2.3 (192.168.2.3), Dst Addr: 192.168.2.2 (192.168.2.2)
Internet Control Message Protocol
Type: 8 (Echo (ping) request)
Code: 0
Checksum: 0x4a5c (correct)
Identifier: 0x0200
Sequence number: 0x0100
Data (32 bytes)
0010 00 3c 00 59 00 00 80 01 b5 12 c0 a8 02 03 c0 a8 .<.Y.... ........
0020 02 02 08 00 4a 5c 02 00 01 00 61 62 63 64 65 66 ....J\.. ..abcdef
0030 67 68 69 6a 6b 6c 6d 6e 6f 70 71 72 73 74 75 76 ghijklmn opqrstuv
0040 77 61 62 63 64 65 66 67 68 69 wabcdefg hi
Sequence number (icmp.seq), 2 bytes
P: 13 D: 13 M: 0

图 5-10

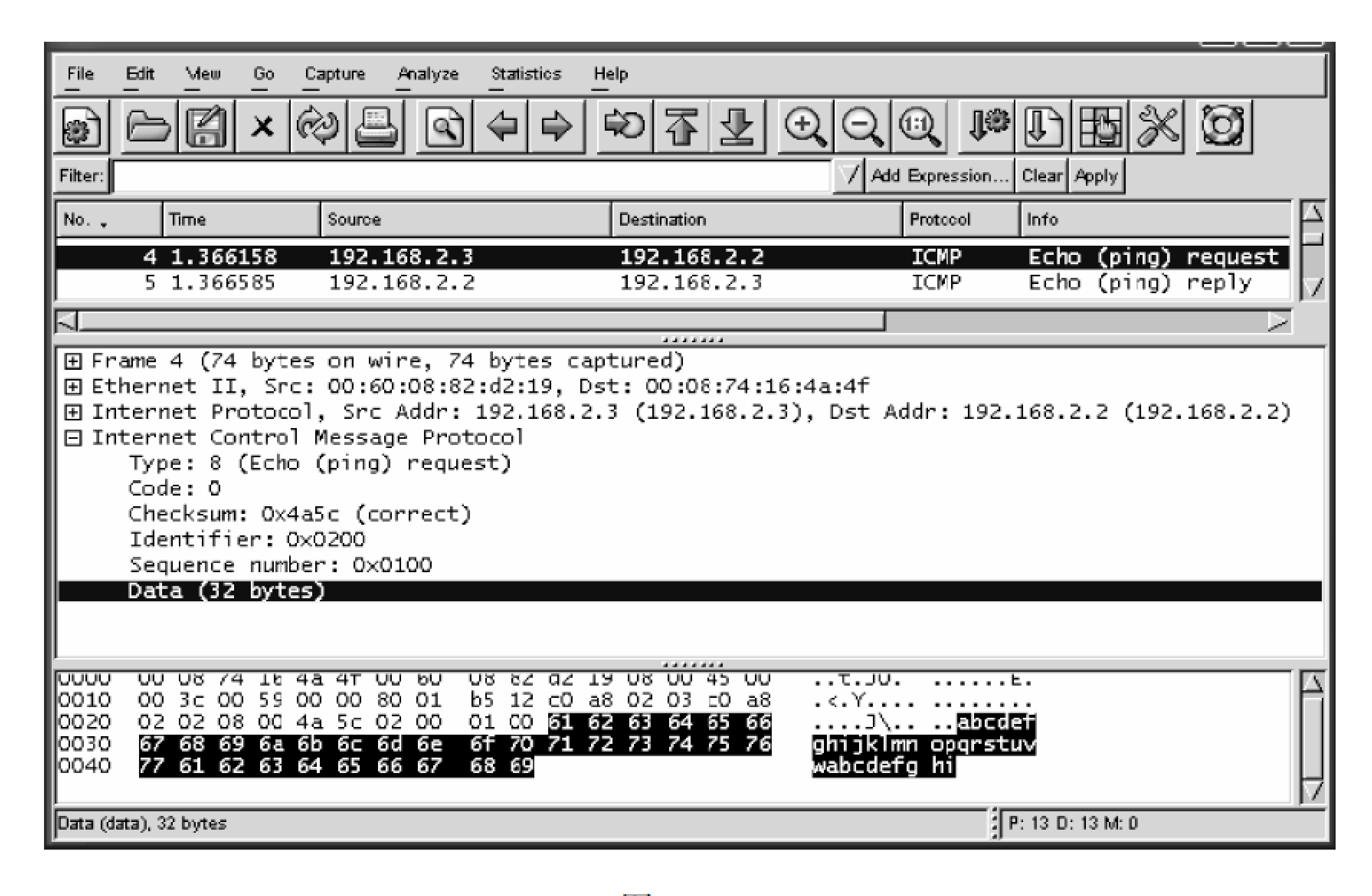

File Edit View Go Capture Analyze Statistics Help
Filter: Add Expression... Clear Apply
No. Time Source Destination Protocol Info
4 1.366158 192.168.2.3 192.168.2.2 ICMP Echo (ping) request
5 1.366585 192.168.2.2 192.168.2.3 ICMP Echo (ping) reply
Frame 4 (74 bytes on wire, 74 bytes captured)
Ethernet II, Src: 00:60:08:82:d2:19, Dst: 00:08:74:16:4a:4f
Internet Protocol, Src Addr: 192.168.2.3 (192.168.2.3), Dst Addr: 192.168.2.2 (192.168.2.2)
Internet Control Message Protocol
Type: 8 (Echo (ping) request)
Code: 0
Checksum: 0x4a5c (correct)
Identifier: 0x0200
Sequence number: 0x0100
Data (32 bytes)
0010 00 3c 00 59 00 00 80 01 b5 12 c0 a8 02 03 c0 a8 .<.Y.... ........
0020 02 02 08 00 4a 5c 02 00 01 00 61 62 63 64 65 66 ....J\.. ..abcdef
0030 67 68 69 6a 6b 6c 6d 6e 6f 70 71 72 73 74 75 76 ghijklmn opqrstuv
0040 77 61 62 63 64 65 66 67 68 69 wabcdefg hi
Data (data), 32 bytes
P: 13 D: 13 M: 0

图 5-11

CHAPTER 6

# 第6章 HTTP 协议

大家对于 HTTP 一定不陌生，平时上网浏览网页，接触最多的可能就是 HTTP 协议了。浏览器向服务器发送请求，服务器回应相应的网页。HTTP 协议从 1990 年开始出现，发展到当前的 HTTP 1.1 标准，已经有了很多的扩展，然而 HTTP 最基本的实现是非常简单的，服务器需要进行的额外处理很少。HTTP 协议使用 TCP 端口 80 来和客户端建立连接。

HTTP 协议是基于请求/响应模式，相当于客户端/服务器模式。客户端向服务器发送一个请求报文，服务器发送响应报文。请求报文/响应报文结构如图 6-1 所示，请求报文包括请求行、报头、空白行、正文，其中请求行是请求报文中的重要部分，请求行包含请求的方法（如表 6-1 所示）、Web 服务器的 URL 和协议版本。请求方法是浏览器发送给服务器的命令，服务器必须按照这些命令来操作，为客户提供服务。响应报文包括响应行、报头、空白行、正文，其中响应行由 HTTP 版本、响应码（如表 6-2 所示）、响应短语和空格组成。正文可以在请求报文或响应报文中，它包含要发送的或接收的文档。空行的作用是把正文和报头隔开。

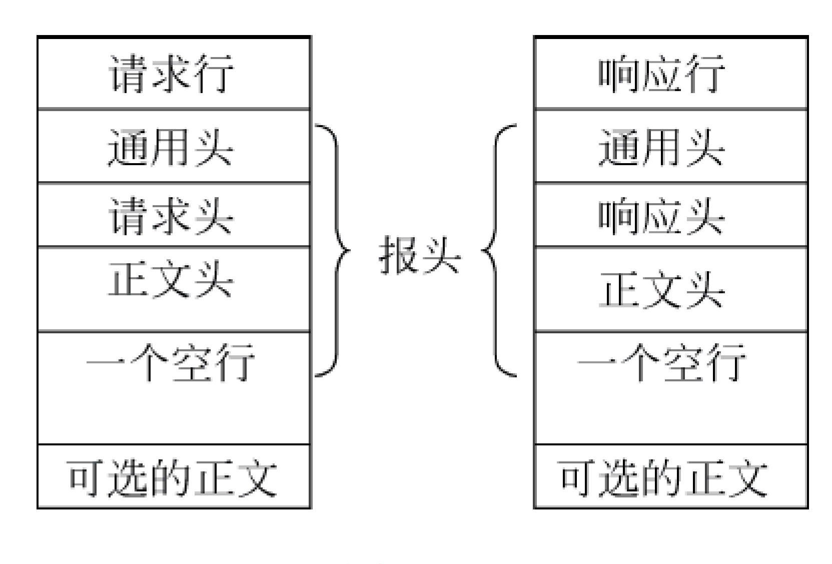



图 6-1

**表 6-1 请求方法**

| 方 法 | 说 明 |
|---|---|
| GET | 当浏览器要从服务器读取指定的文档。GET 方法要求服务器将 URL 定位的资源放在响应报文的正文中，回送给浏览器 |
| HEAD | 当浏览器要从服务器中读取关于文档的首部信息，而不是文档的正文 |
| POST | 从客户端向服务器传送数据，在要求服务器和 CGI 做进一步处理时会用到 POST 方法。POST 主要用于发送 HTML 文本中 FORM 的内容，让 CGI 程序处理。使用 Post 请求时需要在报文首部 Content-Length 字段中指出 body 的长度 |
| PUT | 用从客户端传送的数据取代指定文档的内容 |
| DELETE | 请求服务器删除指定页面 |

表 6-2 常见的响应码

| 响应代码 | 响应信息 | 含　　义 |
|---|---|---|
| 100 | Continue | 初始的请求已经接受,客户应当继续发送请求的其余部分(HTTP 1.1) |
| 101 | Switching Protocols | 服务器将遵从客户的请求转换到另外一种协议(HTTP 1.1) |
| 200 | OK | 一切正常,对 GET 和 POST 请求的应答文档跟在后面 |
| 201 | Created | 服务器已经创建了文档,Location 头给出了它的 URL |
| 202 | Accepted | 已经接受请求,但处理尚未完成 |
| 203 | Non-Authoritative Information | 文档已经正常地返回,但一些应答头可能不正确,因为使用的是文档的拷贝(HTTP 1.1) |
| 204 | No Content | 没有新文档,浏览器应该继续显示原来的文档。如果用户定期地刷新页面,而 Servlet 可以确定用户文档足够新,这个状态代码是很有用的 |
| 205 | Reset Content | 没有新的内容,但浏览器应该重置它所显示的内容。用来强制浏览器清除表单输入内容(HTTP 1.1) |
| 206 | Partial Content | 客户发送了一个带有 Range 头的 GET 请求,服务器完成了它(HTTP 1.1) |
| 400 | Bad Request | 请求出现语法错误 |
| 401 | Unauthorized | 客户试图未经授权访问受密码保护的页面。应答中会包含一个 WWW-Authenticate 头,浏览器据此显示用户名字/密码对话框,然后在填写合适的 Authorization 头后再次发出请求 |
| 403 | Forbidden | 资源不可用。服务器理解客户的请求,但拒绝处理它。通常由于服务器上文件或目录的权限设置导致 |
| 404 | Not Found | 无法找到指定位置的资源。这也是一个常用的应答 |
| 405 | Method Not Allowed | 请求方法(GET、POST、HEAD、DELETE、PUT 和 TRACE 等)对指定的资源不适用(HTTP 1.1) |

接下来讲讲图 6-1 中的报头中各字段用法。

- 通用头包含请求和响应消息都支持的头域,通用头包含 cache-control、connection、date、pragma、transfer-encoding、upgrade 和 mime-version。
- 请求头信息是可选项,请求头只能出现在请求报文中,它用于客户端向服务器提供的客户的配置和客户优先使用的文档格式,常见的请求头信息如表 6-3 所示。

表 6-3 请求头字段

| 信　　息 | 说　　明 |
|---|---|
| Accept | 客户端接受的数据类型 |
| Authorization | 认证消息,包括用户名和口令 |
| User-agent | 客户端软件类型 |
| Accept-charset | 客户端浏览器能够处理的字符集 |
| Accept-encoding | 浏览器能够进行解码的数据编码方式,比如 gzip。Servlet 能够向支持 gzip 的浏览器返回经 gzip 编码的 HTML 页面。许多情形下这可以减少 5～10 倍的下载时间 |
| Accept-language | 浏览器所希望的语言种类,当服务器能够提供一种以上的语言版本时要用到 |

续表

| 信　息 | 说　明 |
| --- | --- |
| Authorization | 授权信息，通常出现在对服务器发送的 WWW-Authenticate 头的应答中 |
| Host | 客户端指定请求初始 URL 中的主机和端口 |
| Range | 客户端请求正文的范围，以字节为单位 |
| IF-Modified-Since | 在 GET 请求中，此字段表示指定日期以来的资源是否被修改，如果修改了，则发送文档。如果没修改，服务器不用发送该文档，返回一个 304 响应代码 |
| User-Agent | 客户端产生请求的软件类型。 |
| Referer | 包含一个 URL，用户从该 URL 代表的页面出发访问当前请求的页面 |
| Cookie | 这是最重要的请求头信息之一 |

- 响应头只能出现在响应报文中，用来向客户端提供服务器的配置信息和关于请求文档的信息，表 6-4 是响应头出现的字段和意义。

**表 6-4　响应头字段**

| 信　息 | 说　明 |
| --- | --- |
| Accept-range | 给出服务器接受客户请求的范围 |
| Age | 给出文档的使用期限 |
| Public | 给出可以支持的方法清单 |
| Retry-afer | 指明的日期之后，服务器才能够使用 |
| Server | 指出服务器程序类型与版本号 |
| ETag | 它向被发送的资源分派一个唯一的标识符。对于可以使用多种 URL 请求的资源，ETag 可以用于确定实际被发送的资源是否为同一资源。例如：ETag：'208f-419e-30f8dc99' |
| Location | 对于一个已经移动的资源，用于重定向请求者至另一个位置 |

- 正文头主要出现在响应报文中，用于说明文档正文的信息，它包含 Allow、Content-encoding、Content-language、Content-length、Content-range、Content-type、Etag、Expires、Last-modified 和 Location 等信息，如表 6-5 所示。

**表 6-5　正文头字段**

| 信　息 | 说　明 |
| --- | --- |
| Allow | 它定义一个由位于请求 URI 中的次源所支持的 HTTP 方法列表。例如：Allow：GET，PUT |
| Content-encoding | 标明一个正文是怎么样编码的 |
| Content-language | 指定正文的自然语言类型 |
| Content-length | 指定包含于请求或响应中数据的字节长度 |
| Content-range | 标明被插入字节的低位与高位字节偏移，也标明此实体的总长度 |
| Content-type | 标明发送或者接收的正文的 MIME 类型。例如：Content-Type：text/html |
| Expires | 指定正文的有效期 |
| Last-modified | 被请求资源上次被修改的日期和时间 |

从上面的介绍可以知道，HTTP 协议通过定义一系列的请求方法、响应状态码和报头字段，来完成服务器和客户端直接的请求和回应，从而完成服务器端向客户端传送 Web 页面的内容。HTTP 传输过程一般分为以下 4 个步骤：

（1）客户端（浏览器）与服务器建立连接。

（2）客户端向服务器请求文档。

（3）服务器响应客户端请求。

（4）断开连接。

如图 6-2 所示，这是个客户端发送的 HTTP 请求报文，客户端发送了 GET 请求，在这个请求行中，客户端的请求方法为 GET，客户端请求服务器的 URL 和协议版本 HTTP1.1。在请求头中，可以看到 Accept、Accept-Language 等字段。通用头部 Connection 表示是否需要持久连接。

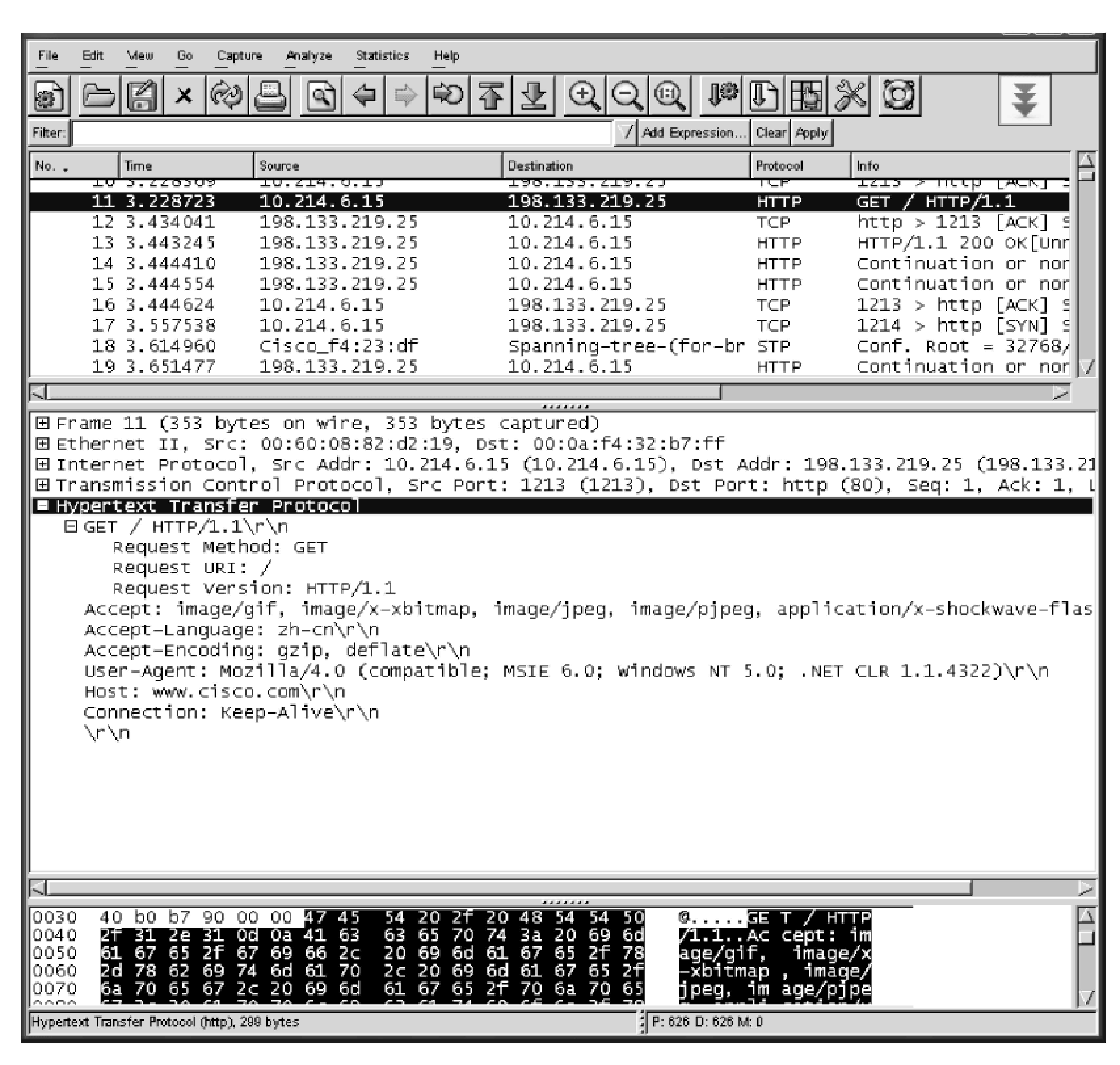



图 6-2

如图 6-3 所示，这是个服务器发送的 HTTP 响应报文，这个报文开始是响应行，它包括 HTTP 版本、响应码 200 和响应短语 OK。Date 响应头字段给出服务器上的时间和日期，通常是格林尼治时间。Server 响应头字段指明了服务器运行的程序类型与版本号。这个报文中 connection 后面有个空行，把报文头部和正文隔开，标明空行的后面就是正文数据了。正文的格式和长度在正文头中定义了，Content-Length：12310，Content-Type：text/css。

HTTP 传输的 4 个步骤通过实例 HTTP 会话来看看这个过程：

在浏览器中键入 www.cisco.com，打开如图 6-4 所示的网页来分析一个 HTTP 的会话过程。

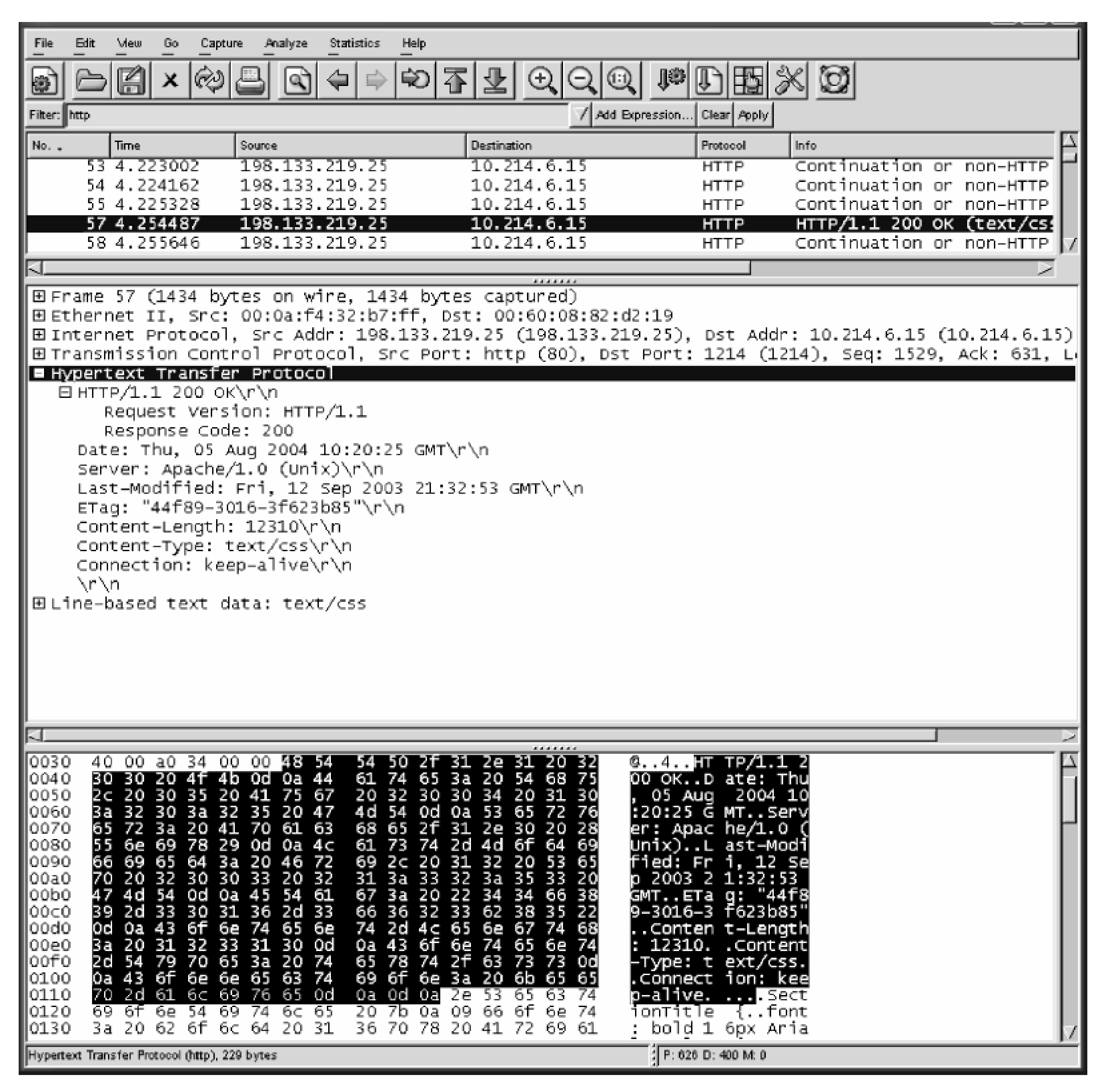



图　6-3




图　6-4

(1) 通过 TCP 三次握手,建立一个 TCP 连接,如图 6-5 所示。

(2) TCP 建立成功后,客户端发送了一个 HTTP 请求,如图 6-6 所示。

(3) 收到客户端的请求,服务器端先发送一个 TCP 确认信息,表示收到了 HTTP 请求,如图 6-7 所示,然后,服务器响应客户端的请求,发送响应的数据,如图 6-8 所示,第 13～第 15 行就是服务器发送的数据。

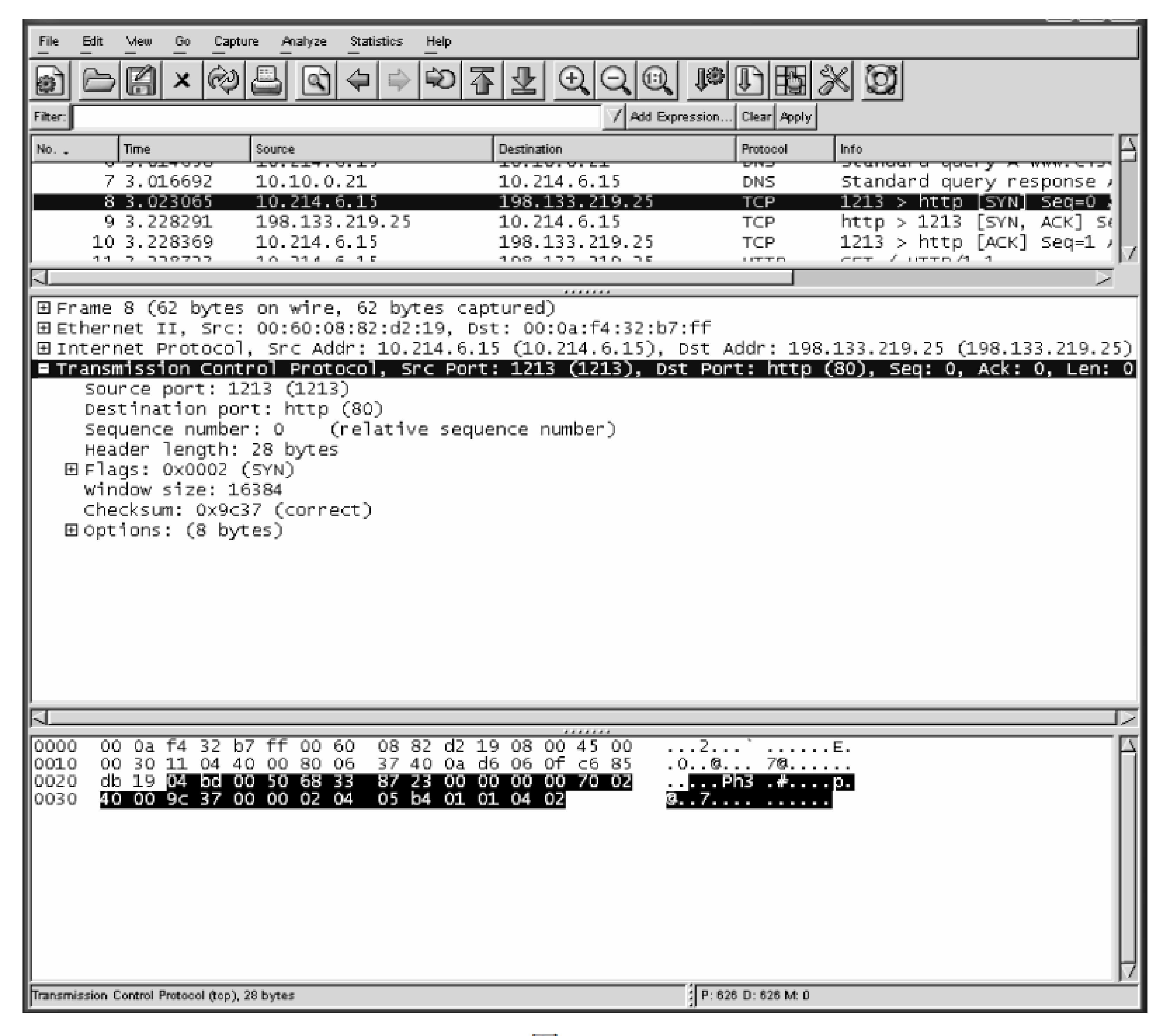

File Edit View Go Capture Analyze Statistics Help
Filter:
Add Expression... Clear Apply
No. Time Source Destination Protocol Info
7 3.016692 10.10.0.21 10.214.6.15 DNS Standard query response
8 3.023065 10.214.6.15 198.133.219.25 TCP 1213 > http [SYN] Seq=0
9 3.228291 198.133.219.25 10.214.6.15 TCP http > 1213 [SYN, ACK]
10 3.228369 10.214.6.15 198.133.219.25 TCP 1213 > http [ACK] Seq=1
Frame 8 (62 bytes on wire, 62 bytes captured)
Ethernet II, Src: 00:60:08:82:d2:19, Dst: 00:0a:f4:32:b7:ff
Internet Protocol, Src Addr: 10.214.6.15 (10.214.6.15), Dst Addr: 198.133.219.25 (198.133.219.25)
Transmission Control Protocol, Src Port: 1213 (1213), Dst Port: http (80), Seq: 0, Ack: 0, Len: 0
Source port: 1213 (1213)
Destination port: http (80)
Sequence number: 0 (relative sequence number)
Header length: 28 bytes
Flags: 0x0002 (SYN)
Window size: 16384
Checksum: 0x9c37 (correct)
Options: (8 bytes)
Transmission Control Protocol (tcp), 28 bytes
P: 626 D: 626 M: 0

图 6-5

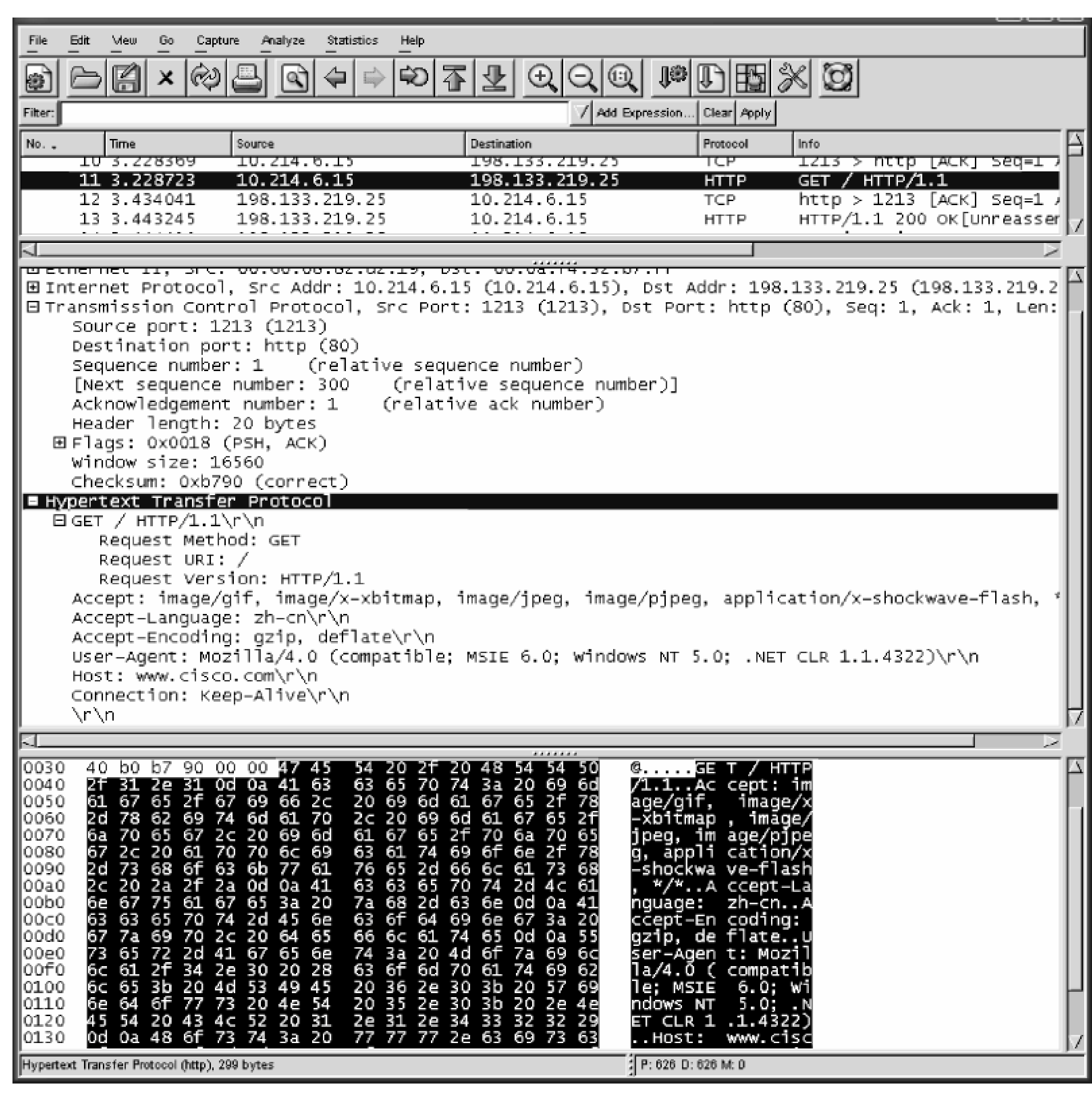

File Edit View Go Capture Analyze Statistics Help
Filter:
Add Expression... Clear Apply
No. Time Source Destination Protocol Info
11 3.228723 10.214.6.15 198.133.219.25 HTTP GET / HTTP/1.1
12 3.434041 198.133.219.25 10.214.6.15 TCP http > 1213 [ACK] Seq=1
13 3.443245 198.133.219.25 10.214.6.15 HTTP HTTP/1.1 200 OK[Unreassem
Internet Protocol, Src Addr: 10.214.6.15 (10.214.6.15), Dst Addr: 198.133.219.25 (198.133.219.2
Transmission Control Protocol, Src Port: 1213 (1213), Dst Port: http (80), Seq: 1, Ack: 1, Len:
Source port: 1213 (1213)
Destination port: http (80)
Sequence number: 1 (relative sequence number)
[Next sequence number: 300 (relative sequence number)]
Acknowledgement number: 1 (relative ack number)
Header length: 20 bytes
Flags: 0x0018 (PSH, ACK)
Window size: 16560
Checksum: 0xb790 (correct)
Hypertext Transfer Protocol
GET / HTTP/1.1\r\n
Request Method: GET
Request URI: /
Request Version: HTTP/1.1
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, application/x-shockwave-flash,
Accept-Language: zh-cn\r\n
Accept-Encoding: gzip, deflate\r\n
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.0; .NET CLR 1.1.4322)\r\n
Host: www.cisco.com\r\n
Connection: Keep-Alive\r\n
\r\n
Hypertext Transfer Protocol (http), 299 bytes
P: 626 D: 626 M: 0

图 6-6

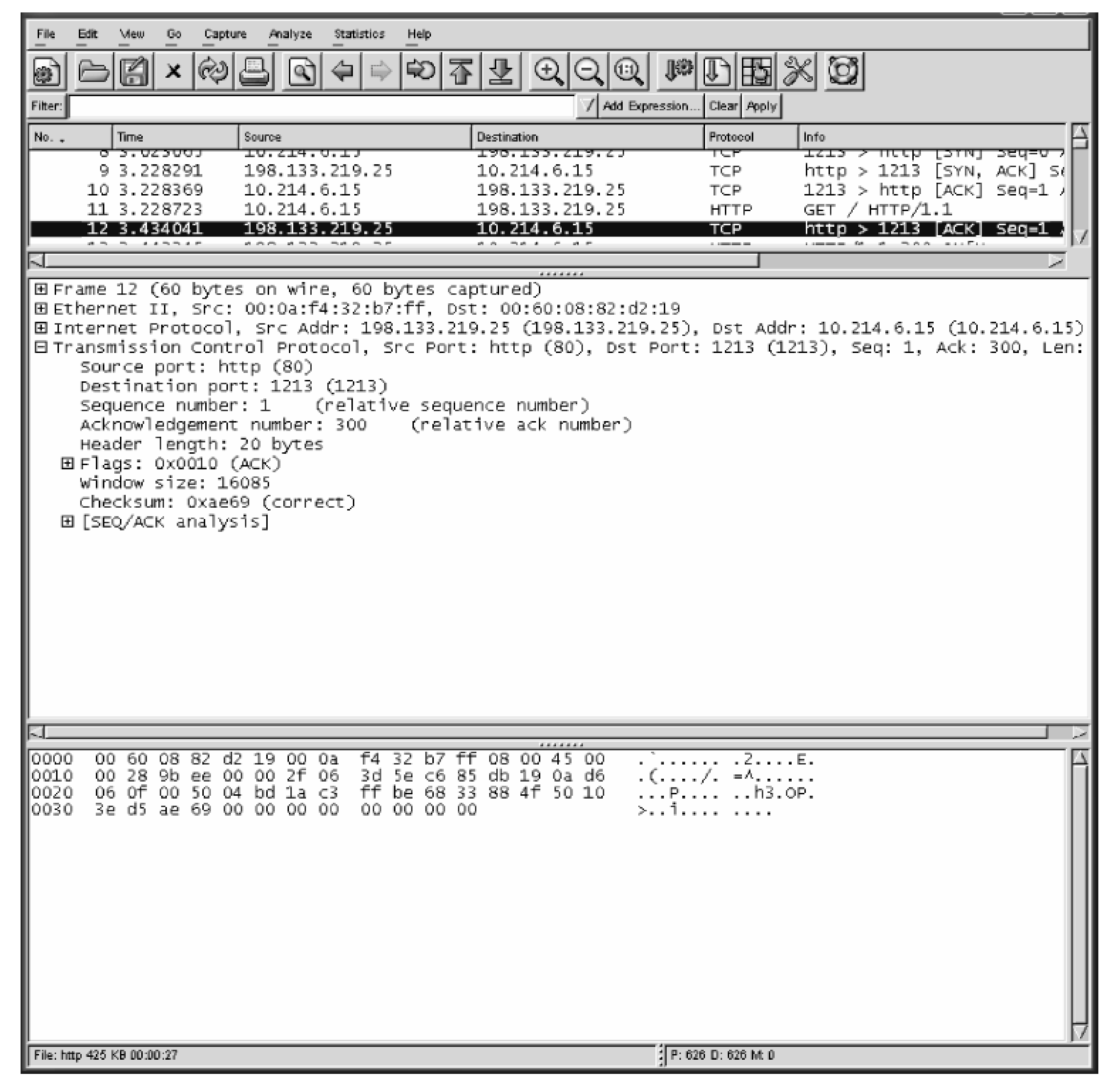


图　6-7

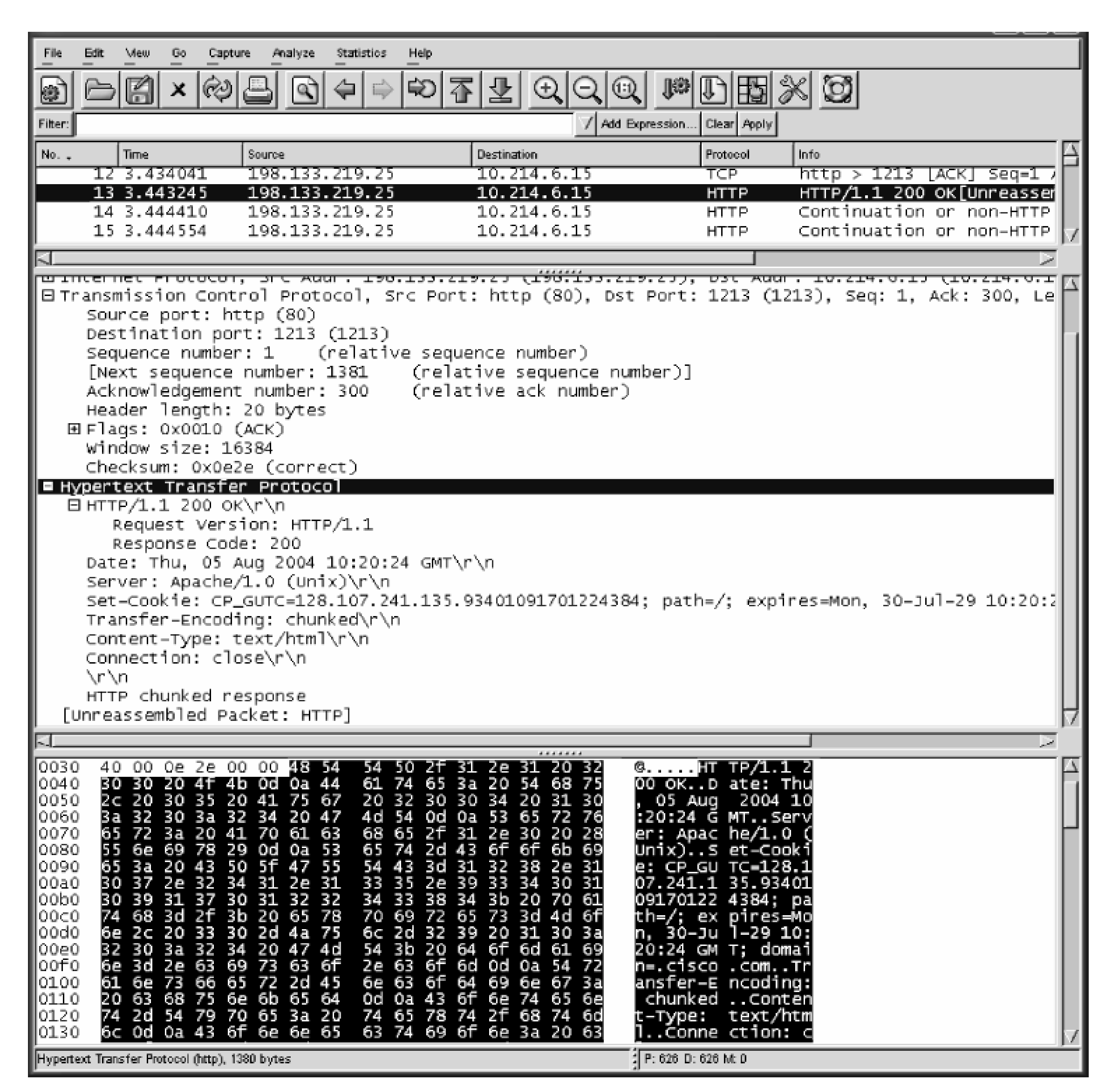


图　6-8

如图 6-9 和图 6-10 所示，服务器发送最后一个数据后，客户端对收到的所有数据进行确认。接下来如果客户端没有其他的请求，服务器就与客户端进行 TCP 4 次握手，断开连接。

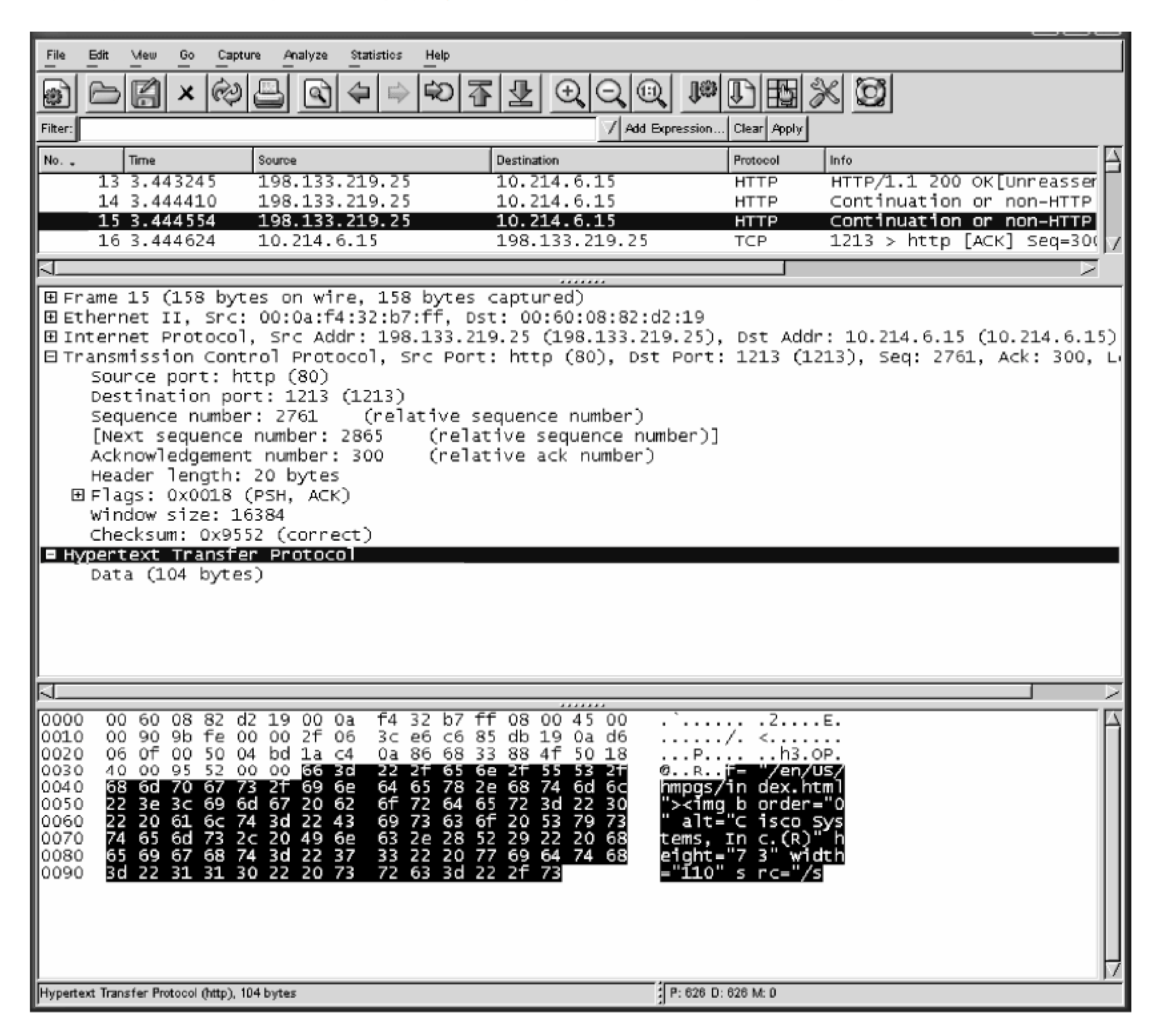



图 6-9

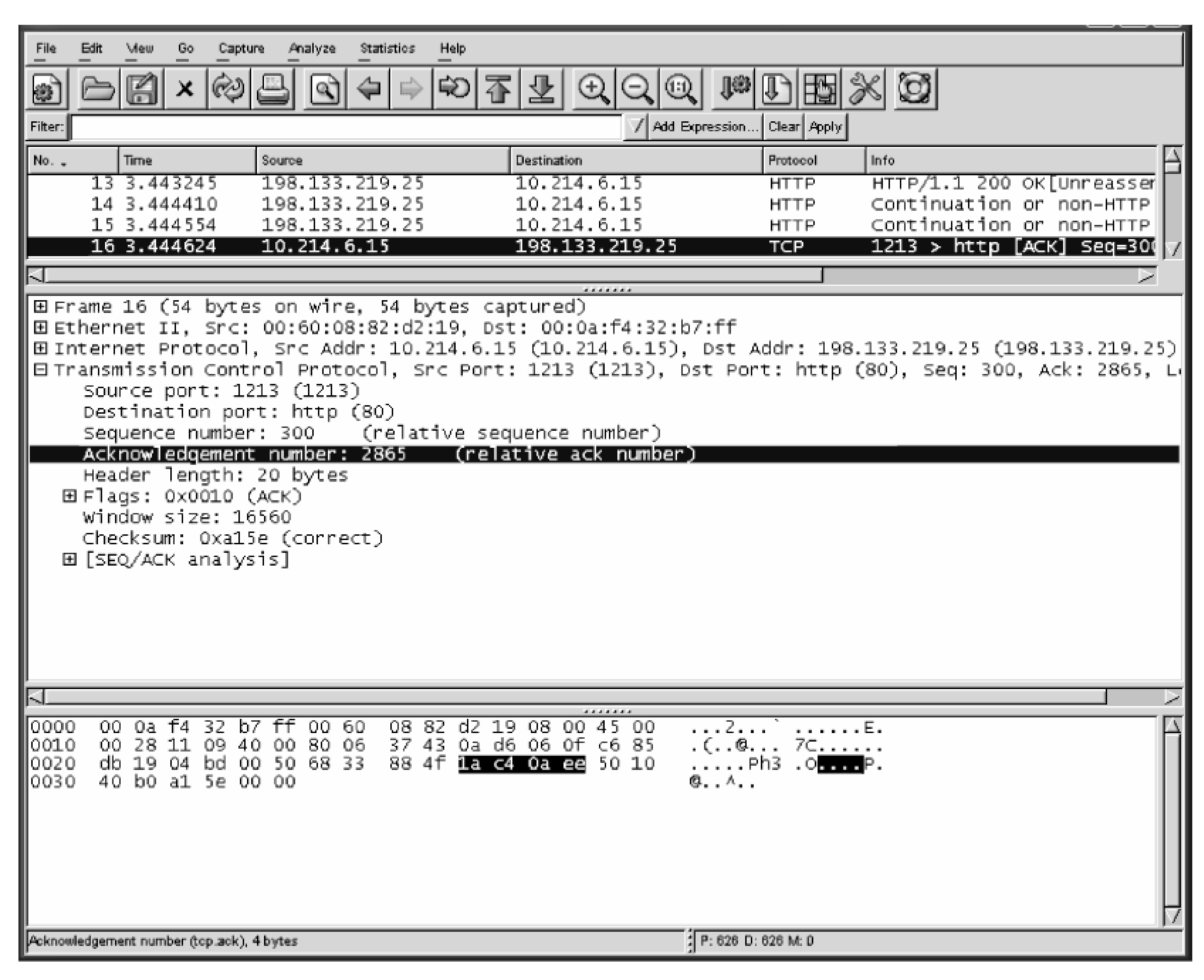



图 6-10

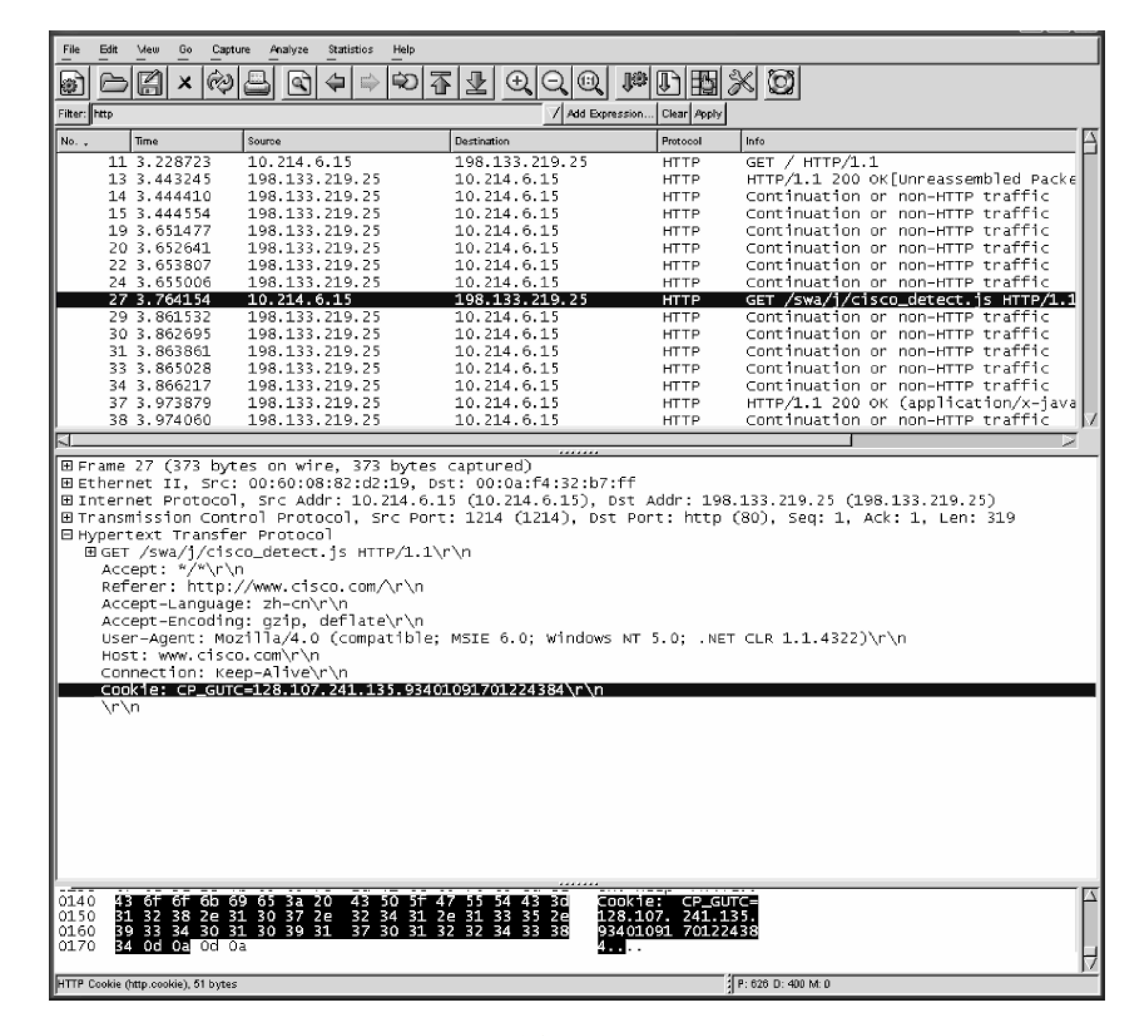

File Edit View Go Capture Analyze Statistics Help
Filter: http
Add Expression... Clear Apply
No. Time Source Destination Protocol Info
11 3.228723 10.214.6.15 198.133.219.25 HTTP GET / HTTP/1.1
13 3.443245 198.133.219.25 10.214.6.15 HTTP HTTP/1.1 200 OK[Unreassembled Packe
14 3.444410 198.133.219.25 10.214.6.15 HTTP Continuation or non-HTTP traffic
15 3.444554 198.133.219.25 10.214.6.15 HTTP Continuation or non-HTTP traffic
19 3.651477 198.133.219.25 10.214.6.15 HTTP Continuation or non-HTTP traffic
20 3.652641 198.133.219.25 10.214.6.15 HTTP Continuation or non-HTTP traffic
22 3.653807 198.133.219.25 10.214.6.15 HTTP Continuation or non-HTTP traffic
24 3.655006 198.133.219.25 10.214.6.15 HTTP Continuation or non-HTTP traffic
27 3.764154 10.214.6.15 198.133.219.25 HTTP GET /swa/j/cisco_detect.js HTTP/1.1
29 3.861532 198.133.219.25 10.214.6.15 HTTP Continuation or non-HTTP traffic
30 3.862695 198.133.219.25 10.214.6.15 HTTP Continuation or non-HTTP traffic
31 3.863861 198.133.219.25 10.214.6.15 HTTP Continuation or non-HTTP traffic
33 3.865028 198.133.219.25 10.214.6.15 HTTP Continuation or non-HTTP traffic
34 3.866217 198.133.219.25 10.214.6.15 HTTP Continuation or non-HTTP traffic
37 3.973879 198.133.219.25 10.214.6.15 HTTP HTTP/1.1 200 OK (application/x-java
38 3.974060 198.133.219.25 10.214.6.15 HTTP Continuation or non-HTTP traffic
Frame 27 (373 bytes on wire, 373 bytes captured)
Ethernet II, Src: 00:60:08:82:d2:19, Dst: 00:0a:f4:32:b7:ff
Internet Protocol, Src Addr: 10.214.6.15 (10.214.6.15), Dst Addr: 198.133.219.25 (198.133.219.25)
Transmission Control Protocol, Src Port: 1214 (1214), Dst Port: http (80), Seq: 1, Ack: 1, Len: 319
Hypertext Transfer Protocol
GET /swa/j/cisco_detect.js HTTP/1.1\r\n
Accept: */*\r\n
Referer: http://www.cisco.com/\r\n
Accept-Language: zh-cn\r\n
Accept-Encoding: gzip, deflate\r\n
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.0; .NET CLR 1.1.4322)\r\n
Host: www.cisco.com\r\n
Connection: Keep-Alive\r\n
Cookie: CP_GUTC=128.107.241.135.93401091701224384\r\n
\r\n
0140 43 6f 6f 6b 69 65 3a 20 43 50 5f 47 55 54 43 3d Cookie: CP_GUTC=
0150 31 32 38 2e 31 30 37 2e 32 34 31 2e 31 33 35 2e 128.107. 241.135.
0160 39 33 34 30 31 30 39 31 37 30 31 32 32 34 33 38 93401091 70122438
0170 34 0d 0a 0d 0a 4....
HTTP Cookie (http.cookie), 51 bytes
P: 626 D: 400 M: 0

图 6-11

CHAPTER 7

# 第 7 章 Telnet 协议

## 7.1 Telnet 协议概述

Telnet(Telecommunication Network Protocol，电信网络协议)起源于1969年的ARPANET，它是一种最老的Internet应用。Telnet使用TCP端口23来提供远程登录，几乎每个TCP/IP的实现都提供这个功能，它能够运行在不同操作系统的主机之间。Telnet通过客户进程和服务器进程之间的选项协商机制，确定通信双方可以提供的功能，如图7-1所示为客户-服务器模式的Telnet。Telnet客户进程和服务器进程一般只是属于用户应用程序，终端用户端通过键盘输入的数据送给操作系统内核的终端驱动进程，由终端驱动进程把用户的输入送到Telnet客户进程，Telnet客户进程把收到的数据传送给TCP，由TCP负责在客户端和服务器端建立TCP连接，数据就通过TCP连接送到了服务器端，服务器的TCP层将收到的数据送到相应的应用层Telnet服务器进程。由于Telnet服务器进程只是应用层程序，不能直接处理来自客户端的(解释或执行)数据，一些执行命令只能通过服务器端的操作系统来完成，因此，Telnet服务器进程把收到的客户端数据通过一个叫“伪终端驱动”送到服务器端的登录Shell进程，同时，Telnet服务器进程也接收服务器内核送到客户端的结果，然后Telnet服务器进程再把收到的结果通

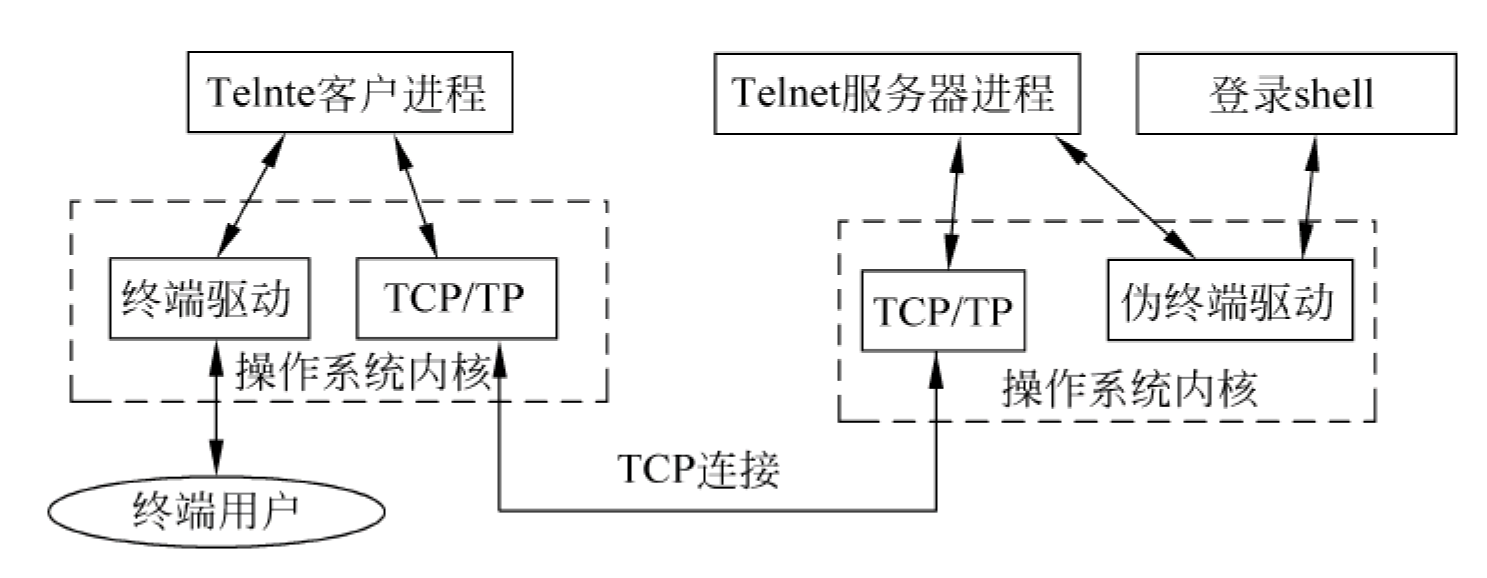



图 7-1

过 TCP 连接传送给 Telnet 客户端进程。"伪终端驱动"可以解释为 Telnet 服务器进程到操作系统内核的接口，负责在 Telnet 服务器进程和登录 Shell 进程之间传送数据，对于登录 Shell 来讲，它直接被 Telnet 服务器进程调用，任何运行在登录 Shell 进程处的程序感觉是直接和一个终端在打交道。

Telnet 可以运行在不同的操作系统之间，那么不同的操作系统之间使用的命令不同，比如一些操作系统需要每行文本用 ASCII 回车控制符(CR)结束，另一些系统则需要使用 ASCII 换行符(LF)，还有一些系统需要用两个字符的序列回车-换行(CR-LF)；再比如，大多数操作系统为用户提供了一个中断程序运行的快捷键，但这个快捷键在各个系统中又有可能不同(一些系统使用 CTRL+C，而另一些系统使用 ESCAPE)。因此，为了让 Telnet 协议适应异构环境，Telnet 定义了 NVT(Net Virtual Terminal)，NVT 定义了数据和命令在 Internet 上的传输方式，如图 7-2 所示。当客户端发送数据时，客户端软件把来自用户终端的键盘输入转换为 NVT 格式发送到服务器端，服务器端软件将收到的数据和命令从 NVT 格式转换为远程系统需要的格式；当服务器端返回数据时，远程服务器端将数据从远程服务器的格式转换为 NVT 格式发送给客户端，客户端将接收到的 NVT 格式数据再转换为本地的格式。

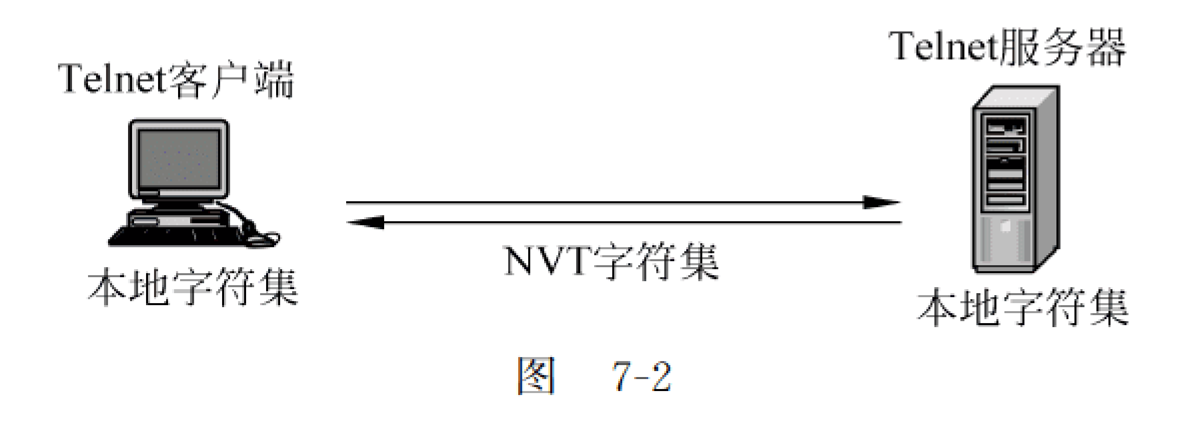



图　7-2

图 7-2 中的数据离开本地后就转换为 NVT 字符集进行传输，NVT 字符集有两种类型：一种是数据字符集，一种是远程控制字符集。当传输数据时，NVT 就用数据字符集即 NVT ASCII，它的是个 8 位字符集，其中最高位是 0，其他低 7 位和 US ASCII 码一致，行结束处用两个字符 CR(回车)和 LF(换行)表示序列结束，用\r\n 来表示。单独的一个 CR 也是以两个字符序列来表示，它们是 CR 和紧接着的 NUL(字节 0)，用\r\0 表示。当传输控制命令时，NVT 就使用 NVT 远程控制字符集，它是一个 8 位字符集，最高位是 1，最常用的远程控制字符集如表 7-1 所示。既然 Telnet 有两种字符集，那么怎么区分发送的是数据字符还是远程控制字符呢？

**表 7-1　NVT 远程控制字符集**

| 字符 | 十进制 | 二进制 | 意　　义 |
|---|---|---|---|
| EOF | 236 | 11101100 | 文件结束 |
| EOR | 239 | 11101111 | 记录结束 |
| SE | 240 | 11110000 | 子选项结束 |
| NOP | 241 | 11110001 | 无操作 |
| DM | 242 | 11110010 | 数据标记 |
| BRK | 243 | 11110011 | 断开 |
| IP | 244 | 11110111 | 中断进程 |
| AO | 245 | 11110101 | 异常终止输出 |
| AYT | 246 | 11110110 | 对方是否还在运行 |
| EC | 247 | 11110111 | 擦除最后一个字符 |

续表

| 字符 | 十进制 | 二进制 | 意　　义 |
|---|---|---|---|
| EL | 248 | 11111000 | 擦除行 |
| GA | 249 | 11111001 | 前进 |
| SB | 250 | 11111010 | 自选项开始 |
| WILL | 251 | 11111011 | 同意激活选项 |
| WONT | 252 | 11111100 | 拒绝激活选项 |
| DO | 253 | 11111101 | 认可选项请求 |
| DONT | 254 | 11111110 | 拒绝选项请求 |
| IAC | 255 | 11111111 | 解释(下一个字符)为控制字符 |

表 7-1 中,有个 GA 命令,Telnet 中的 GA 命令提供了一个机制,使“远程”计算机(服务器)如何给“本地”计算机(用户)发送信号,告诉对方现在是给用户终端传递控制的时间。当用户需要获得对终端的控制时,它应该并且只能在这段时间传递。

Telnet 的服务器进程和客户进程工作的 4 种方式如下。

### 1. 半双工

这是 Telnet 的默认方式,现在已经很少使用了,用户输入的每个字符回显到屏幕,但整个一行完成前客户端并不发送它,在将整个一行发送给服务器后,客户端在接收来自用户输入的一个新行之前,要等待来自服务器的 GA 命令。它不能充分发挥目前大量使用的支持全双工通信的终端功能。

### 2. 一次一字符方式

用户输入一个字符,发送给服务器,服务器确认收到的字符,将该字符回显,除非服务器进程端的应用程序去掉了回显功能,客户确认收到回显的字符。要进入这种方式,只要激活服务器进程的 SUPPRESS GO AHEAD(抑制前进)选项和 ECHO 选项。这可以通过由客户进程发送 DO SUPPRESS GO AHEAD(请求激活服务器的抑制前进选项)请求完成,也可以通过服务器进程给客户进程发送 WILL SUPPRESS GO AHEAD(服务器将激活抑制前进选项)来完成。服务器进程通常还会跟着发送 WILL ECHO,以使回显功能有效。缺省情况下 Telnet 登录时进入字符输入方式。

### 3. 一次一行方式

该方式通常叫做准行方式(kludgelinemode),该方式的实现是遵照 RFC858 的。该 RFC 规定:如果要实现带远程回显的一次一个字符方式,ECHO 选项和 SUPPRESS GO AHEAD 选项必须同时有效。准行方式采用这种方式来表示当两个选项的其中之一无效时,Telnet 就是工作在一次一行方式。

### 4. 行方式

在 RFC1184 中定义。行方式也是通过客户进程和服务器进程进行协商而确定的,它纠正了准行方式的所有缺陷。行方式工作在全双工状态下,行编辑(回显、字符擦除、行擦除等)由客户端来完成。目前比较新的 Telnet 实现支持这种方式。

## 7.2　选项协商

Telnet 进行连接的双方首先进行选项协商，协商 Telnet 工作的一些环境、工作方式等，选项协商是对称的，也就是说任何一方都可以主动发送选项协商请求给对方。对于任何给定的选项，连接的任何一方都可以发送下面 4 种请求的任意一个。

（1）WILL：发送方本身将激活（enable）选项。

（2）DO：发送方想叫接收端激活选项。

（3）WON'T：发送方本身想禁止选项。

（4）DON'T：发送方想让接收端去禁止选项。

由于 Telnet 规则规定，对于激活选项请求（如 1 和 2），有权同意或不同意。而对于使选项失效请求（如 3 和 4），必须同意。这样，4 种请求就会组合出 6 种情况，如表 7-2 所示。

**表　7-2**

| | 发　送　方 | 接　收　方 | 描　　述 |
|---|---|---|---|
| 1 | WILL | DO | 发送方想激活选项<br>接收方说同意 |
| 2 | WILL | DON'T | 发送方想激活选项<br>接收方说不同意 |
| 3 | DO | WILL | 发送方想让接收方激活选项<br>接收方说同意 |
| 4 | DO | WON'T | 发送方想让接收方激活选项<br>接收方说不同意 |
| 5 | WON'T | DON'T | 发送方想禁止选项<br>接收方必须说同意 |
| 6 | DON'T | WON'T | 发送方想让接收方禁止选项<br>接收方必须说同意 |

选项协商需要 3 个字节：一个 IAC 字节，接着一个字节是 WILL、DO、WON'T 或 DON'T 这 4 个中的一个，最后一个 ID 字节指明要激活或禁止的选项。选项协商的常用选项代码如表 7-3 所示。

**表　7-3**

| 十进制 | 十六进制 | 说　　明 |
|---|---|---|
| 0 | 0x00 | 使用 8 位二进制传输 |
| 1 | 0x01 | 回显（echo），将接收到的字符返回给发送者 |
| 3 | 0x03 | 抑制继续前进（字符方式可以选择这个选项） |
| 24 | 0x18 | 终端类型 |
| 31 | 0x1F | 窗口大小 |
| 32 | 0x20 | 终端速率 |
| 33 | 0x21 | 远程流量控制 |
| 34 | 0x22 | 行方式 |
| 35 | 0x23 | X 显示定位 |
| 36 | 0x24 | 环境变量 |

表 7-3 中的终端类型、终端速率等的协商，需要附加的信息，比如终端类型的协商需要附加字符串来表明终端的类型，终端的速率需要附加数字来表明终端的速率，这样需要进一步附加数字或字符串的协商，要用子协商来定义，子协商使用的 NVT 字符集如表 7-4 所示。

**表　7-4**

| 字符 | 十进制 | 意　　义 |
|---|---|---|
| SE | 240 | 子选项结束 |
| SB | 250 | 子选项开始 |

## 7.3　Telnet 报文分析

接下来看看 Telnet 的一次一个字符方式的过程，客户端地址为 10.61.16.11，服务器端地址为 10.214.6.11。

首先客户端和服务器端进行 TCP 三次握手，如图 7-3 所示的 TCP 连接可以看到，客户端 10.61.16.11 使用随机端口 3148 连接 Telnet 服务器的端口 23。

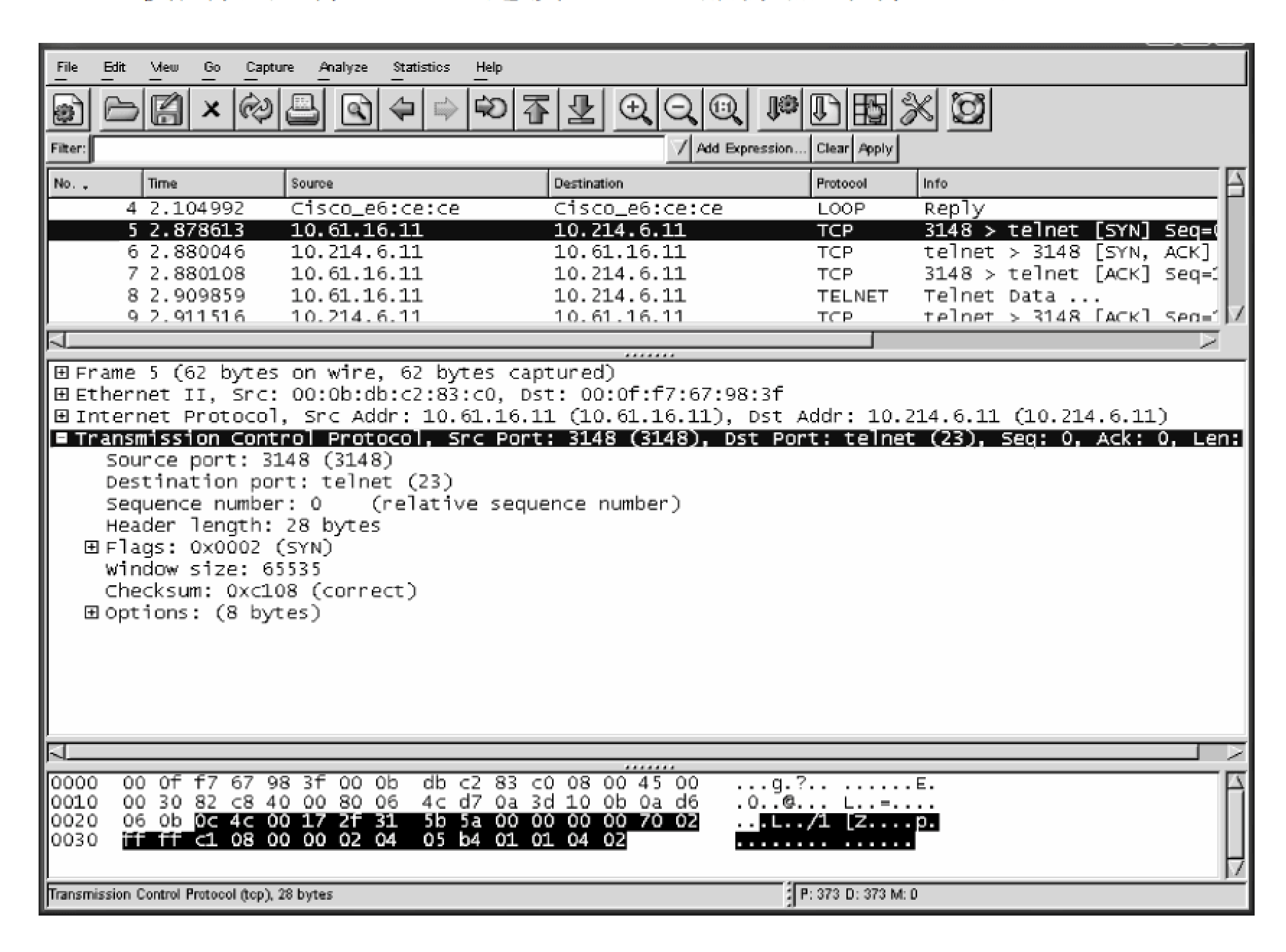



图　7-3

字符方式只要激活服务器进程的 SUPPRESS GO AHEAD（抑制前进）和 ECHO 选项即可，如图 7-4 所示，客户端发起 DO SUPPRESS GO AHEAD 选项协商，表示客户端请求激活“抑制前进”选项。在这个命令中，可以看到十六进制数值：ff fb 03。其中，ff 就是前面讲到的 IAC，fd 是命令 DO 的十六进制代码，03 是 SUPPRESS GO AHEAD 选项的代码。

如图 7-5 所示，服务器端发送 TCP 确认，确认收到图 7-4 的报文。

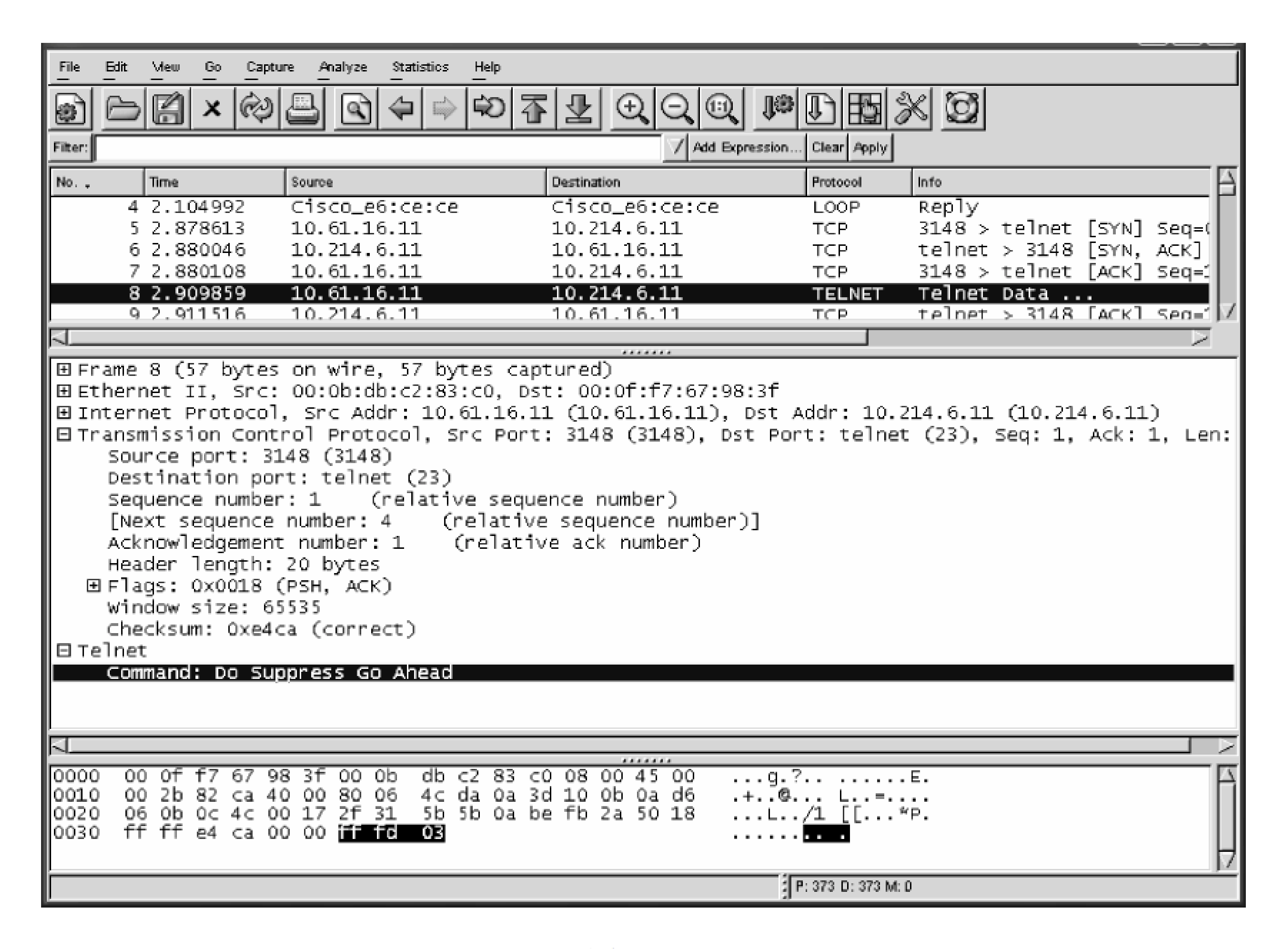



图　7-4

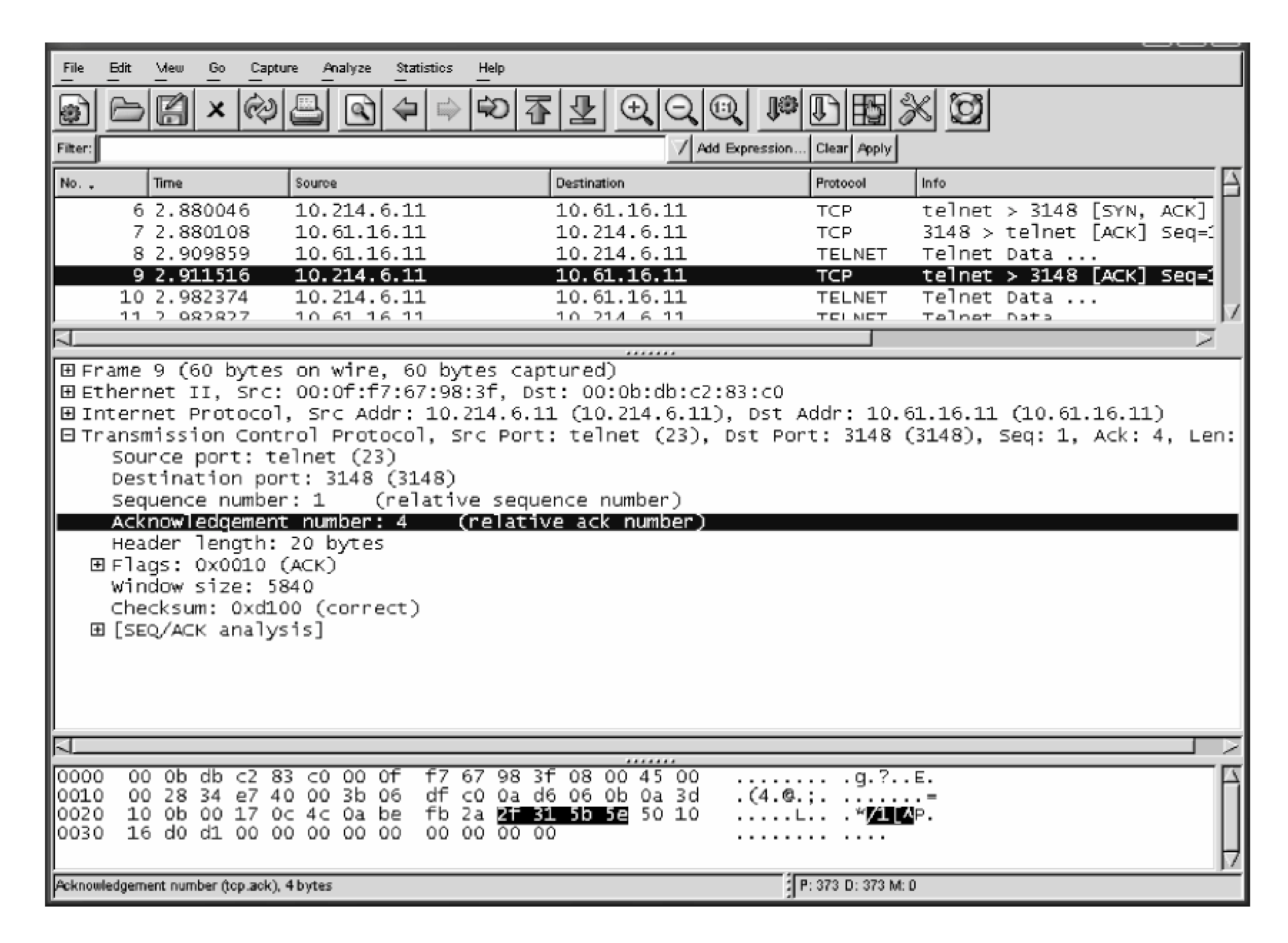



图　7-5

如图 7-6 所示，服务器端发起选项协商，可以看到选项内容：终端类型、终端速度、显示定位选项、环境变量。这里用 DO 表示服务器端希望客户端在子协商中发送关于这些选项协商的数据。图中黑色部分十六进制数 ff fd 18 表示 IAC　DO　TERMINAL TYPE，表明服务器端想在一个后续子协商中接收客户端发送的关于终端类型的信息。

接下来 ff fd 20 表示 IAC　DO　TERMINAL speed，表明服务器端想在一个后续子协商中接收客户端发送的关于终端速率的信息。

ff fd 23 表示 IAC DO X DISPLAY LOCATION，表明服务器端希望在子协商中接受 X 显示定位。

最后是 ff fd 27 表示 IAC DO NEW ENVIRON option，表明服务器端希望接收环境变量参数。

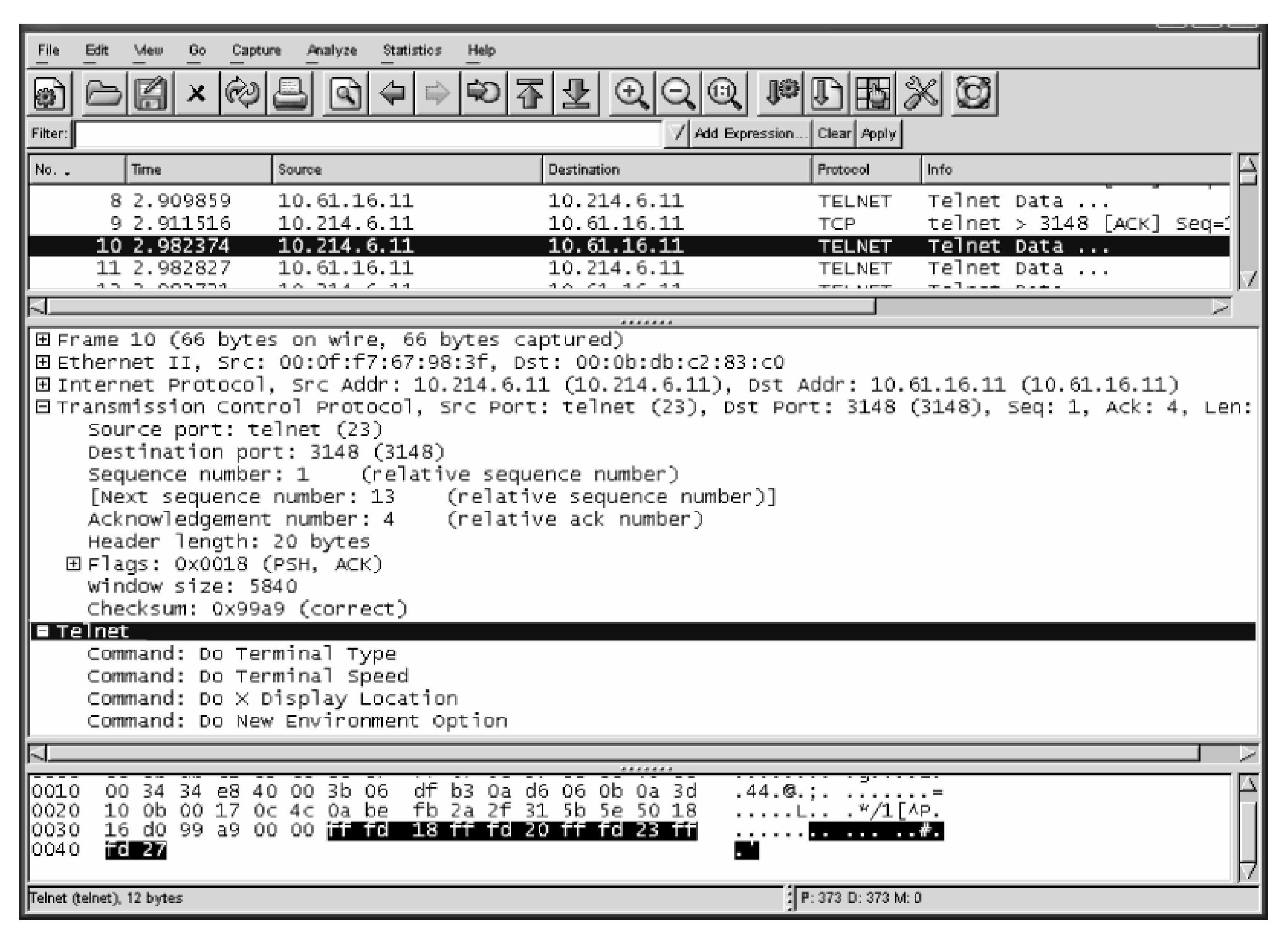



图　7-6

如图 7-7 所示，客户端同意发送终端类型。

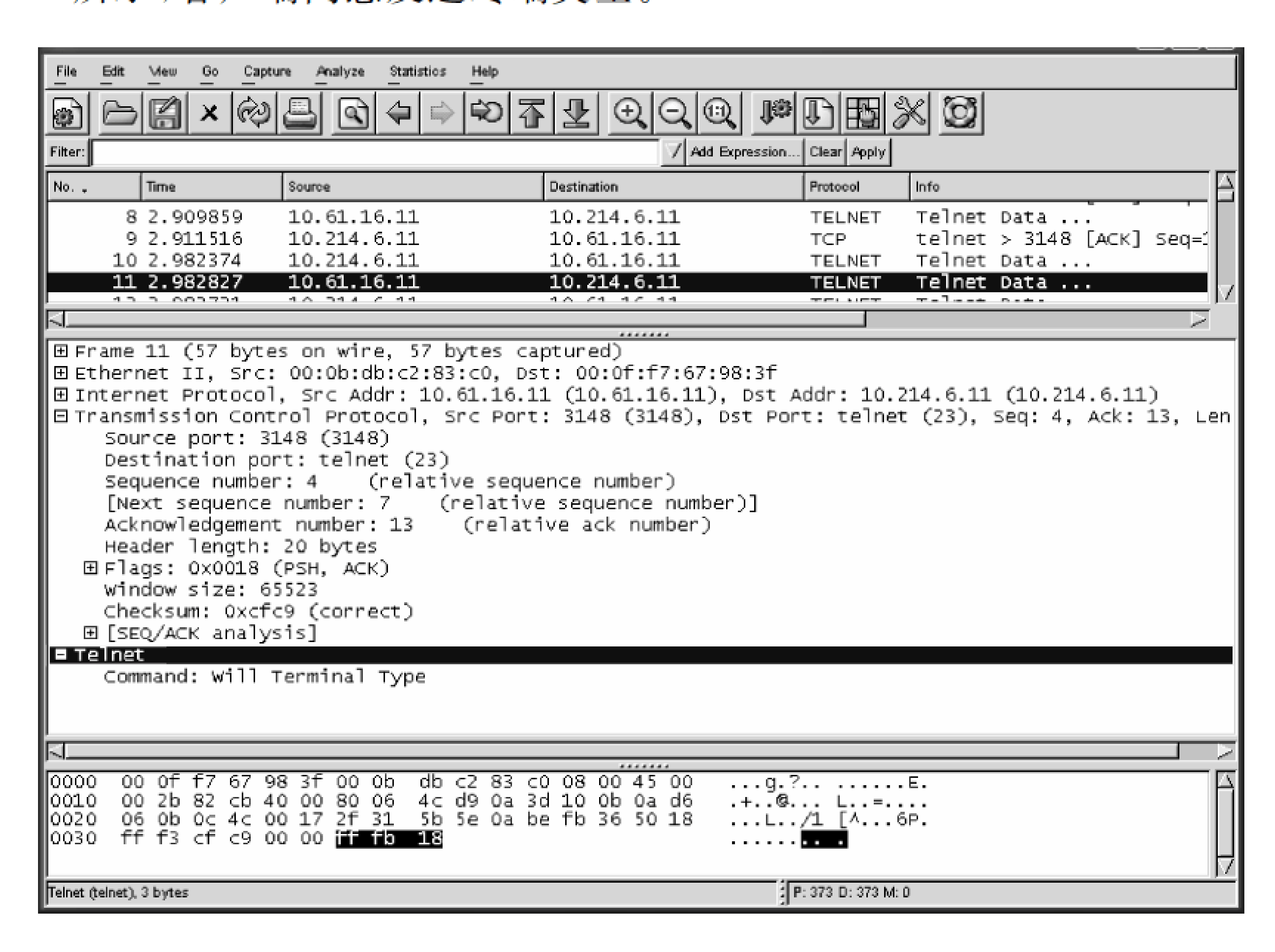



图　7-7

如图 7-8 所示，服务器端同意抑制前进选项。

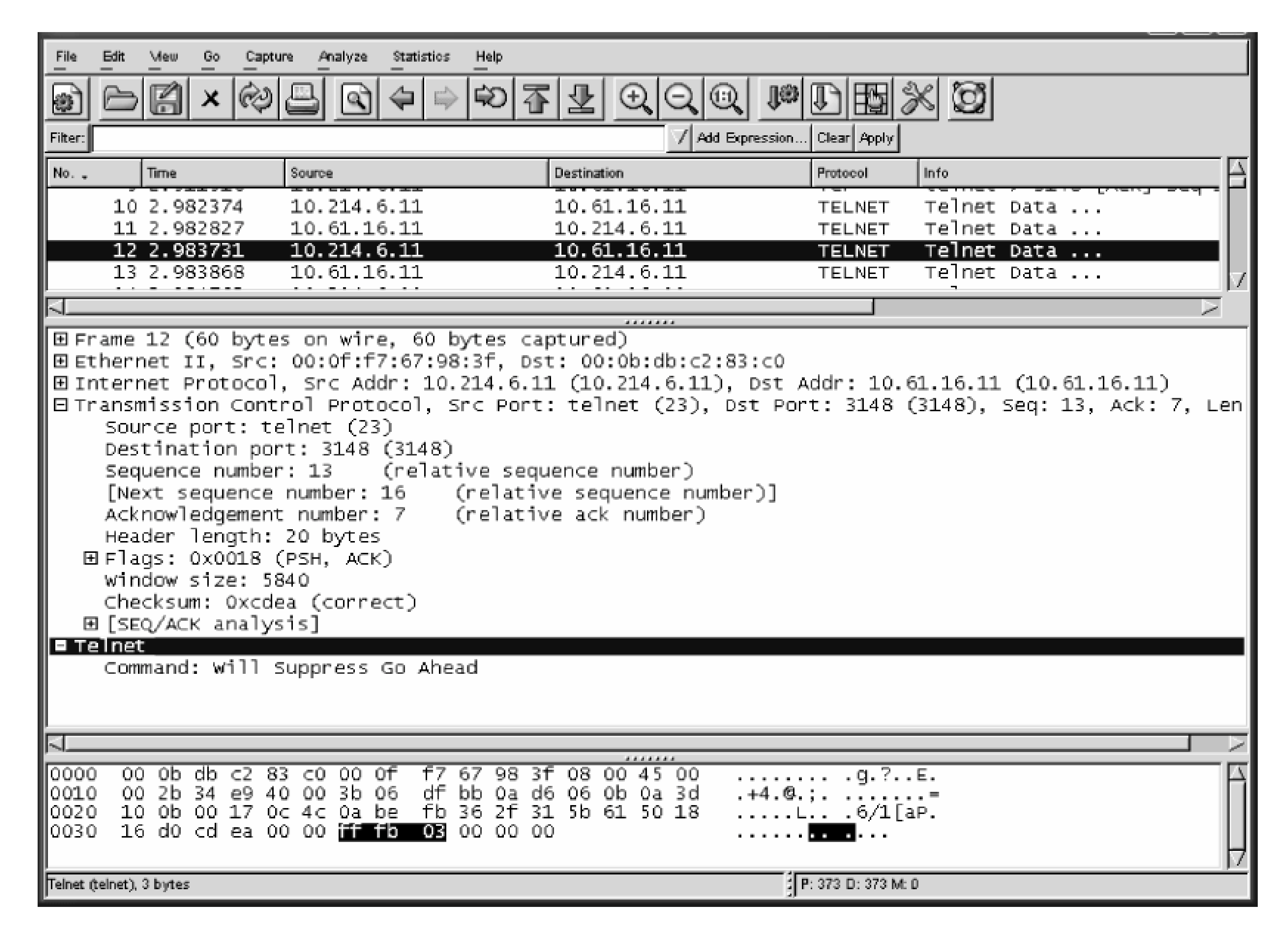



图 7-8

如图 7-9 所示，客户端拒绝对终端速度、显示定位选项、环境变量进行协商，也就是说客户端不会发送关于服务器端请求的跟这些选项有关的数据。

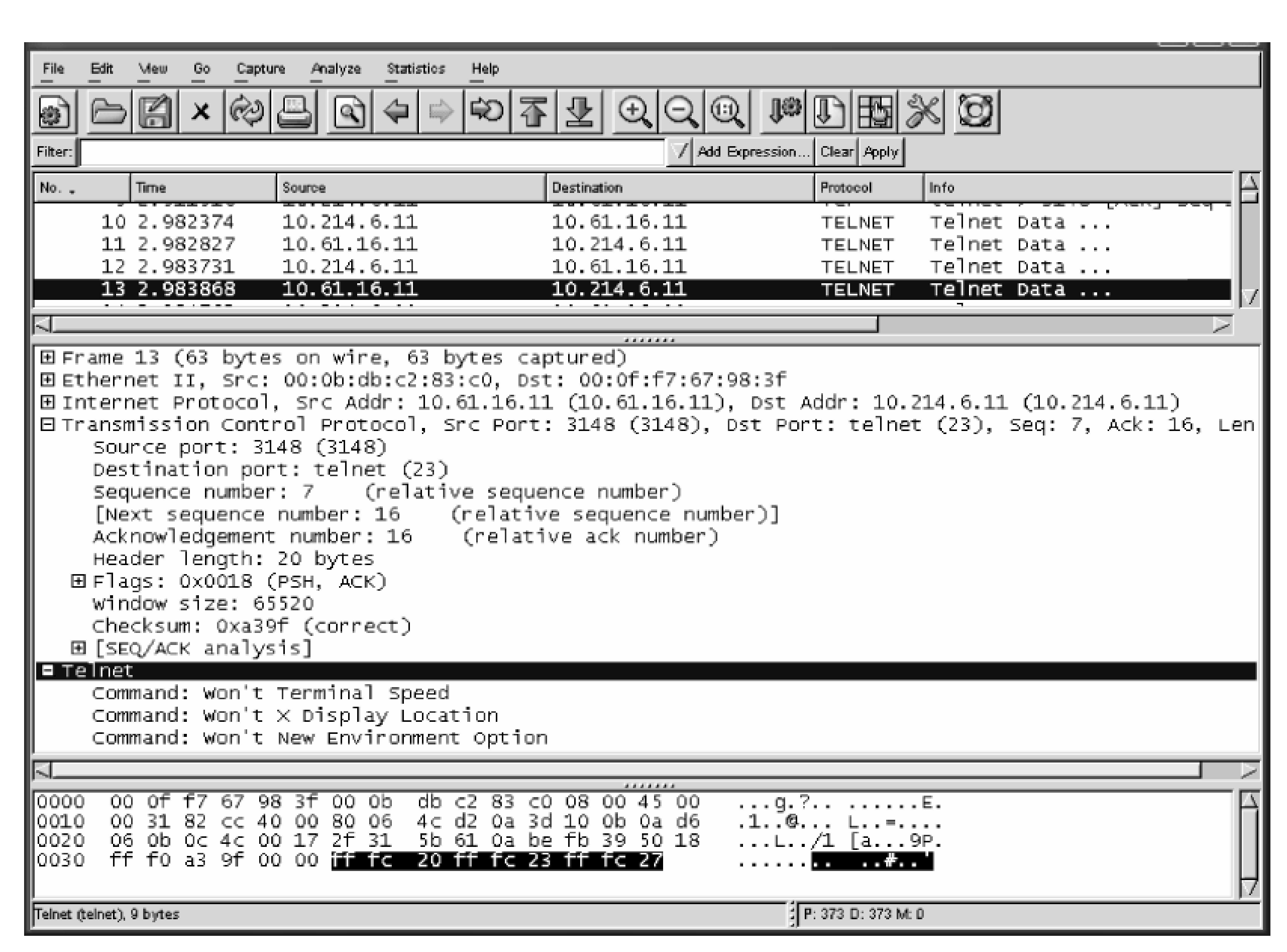



图 7-9

如图 7-10 所示，服务器发起对终端类型进行子协商，可以看到这个子协商用 suboption begin 开始，协商关于是关于终端类型，要求客户端发送终端类型，最后如图 7-11 所示用

Suboption End 结束这个子协商。

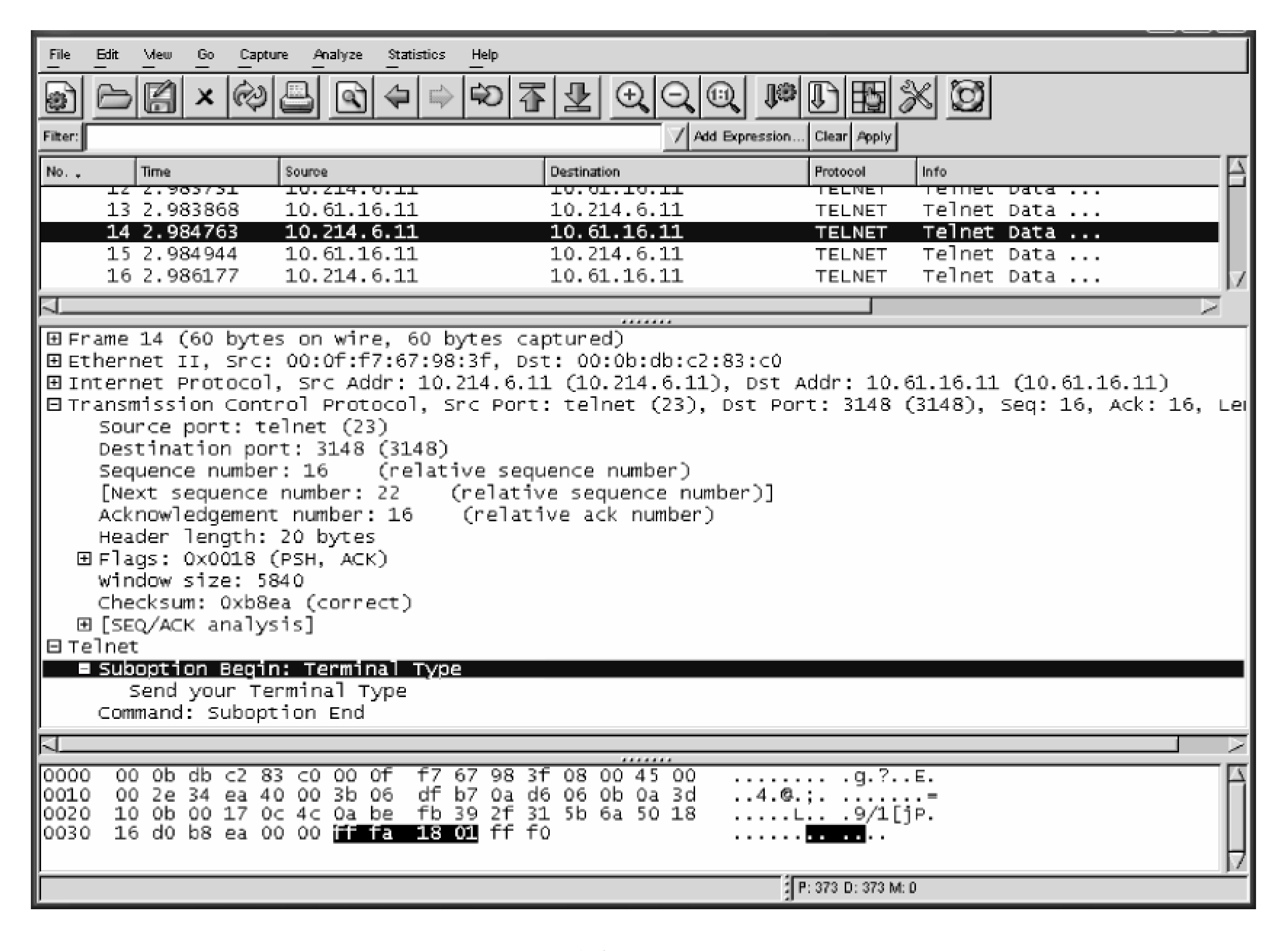



图 7-10

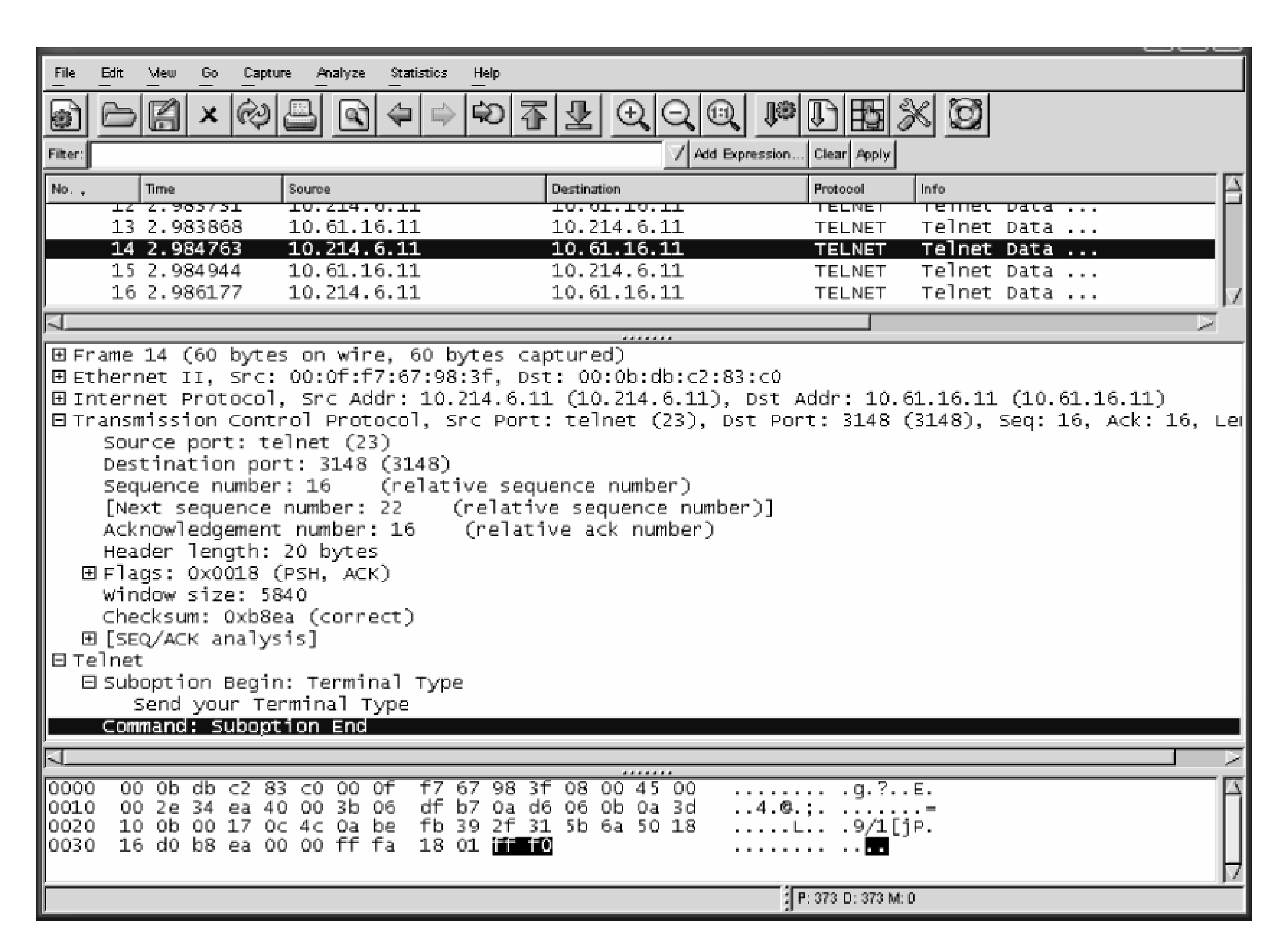



图 7-11

如图 7-12 所示，客户端发送子选项协商终端类型的数值，图中十六进制数 ff fa 18 表示客户端对终端类型的子协商开始，十六进制数 00 表示后续的值用 8 位二进制发送，76 74 31 30 30 对应 ASCII 码 vt100。

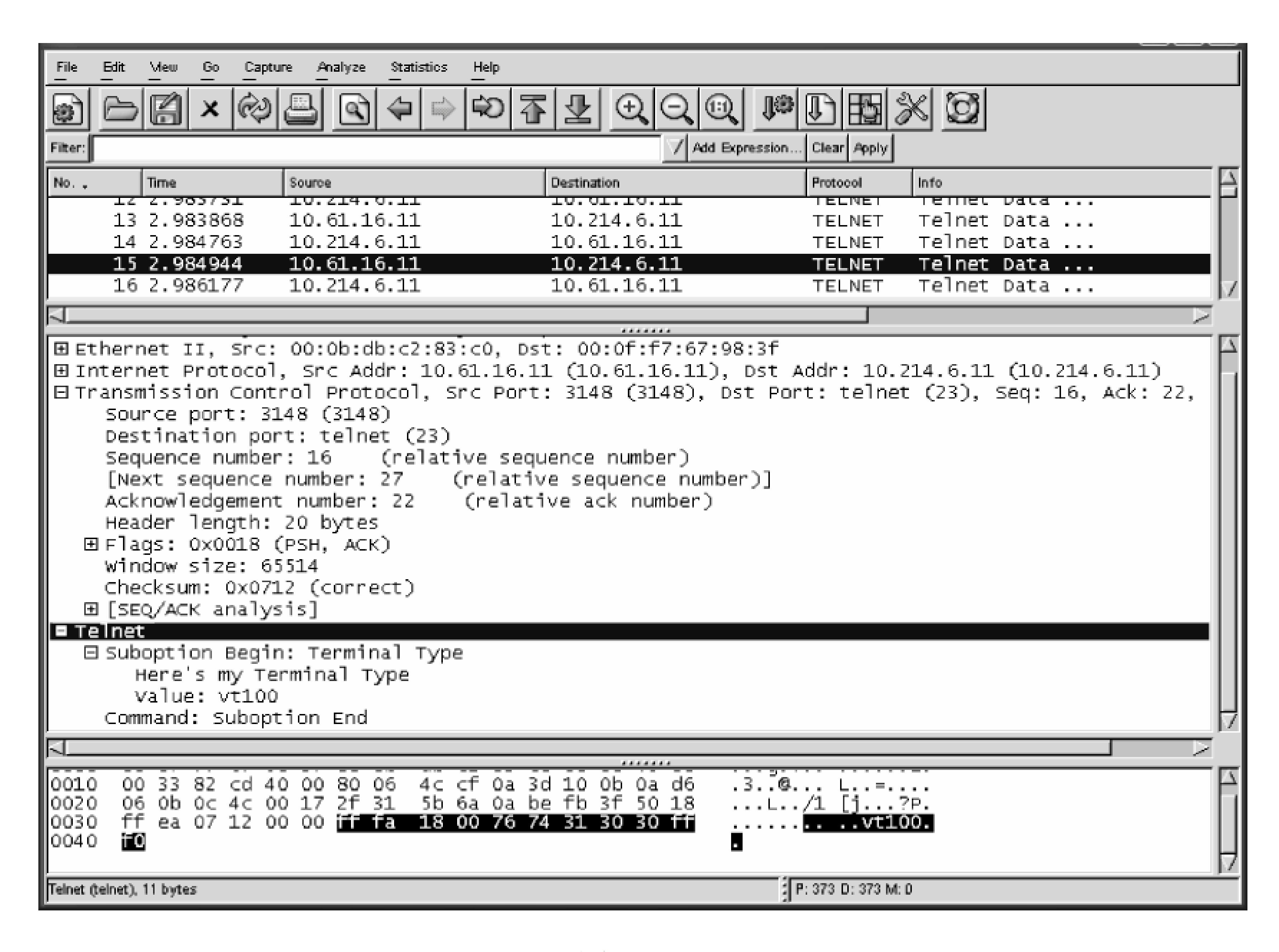



图 7-12

如图 7-13 所示，服务器发起回显、协商窗口的尺寸、状态信息、远程流控制等选项协商。DO 表明服务器请求客户端发送关于选项的数据，如 DO Echo 表明服务器请求客户端回显收到的字符。Will 表明服务器自身想发送关于选项的数据，如 Will status 表明服务器想发送状态信息。

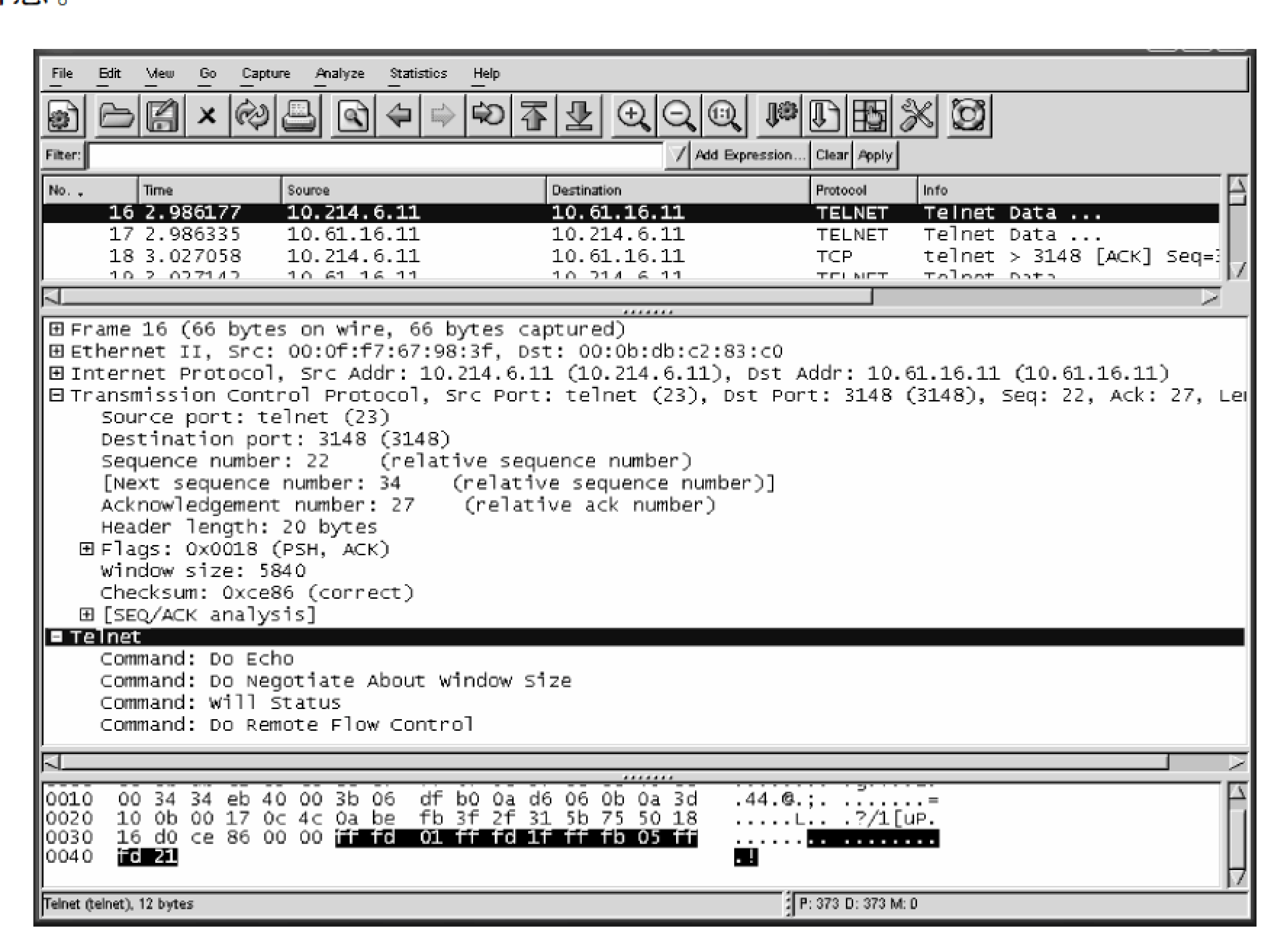



图 7-13

如图 7-14 所示，客户端拒绝回显接收到的字符。

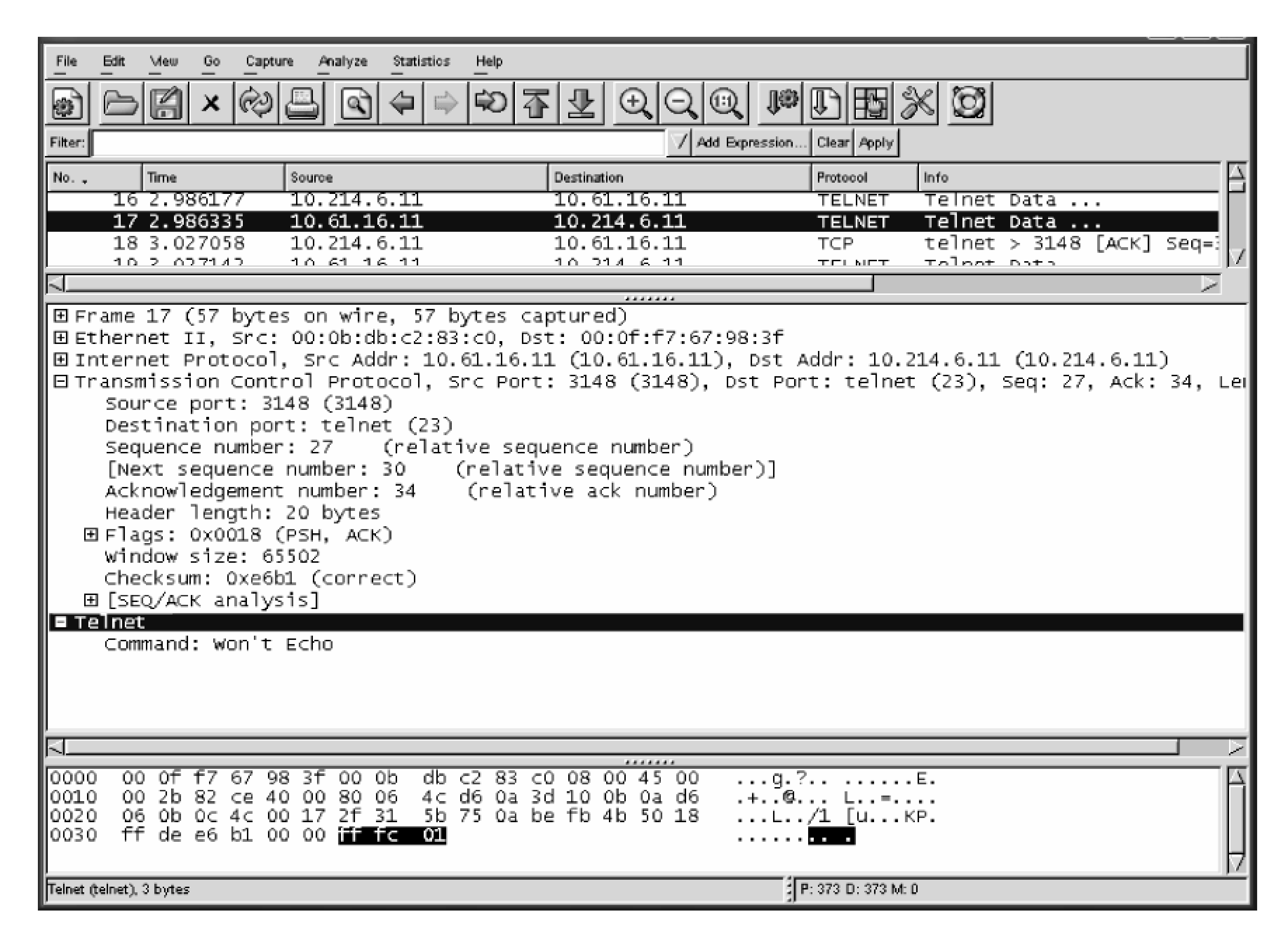



图 7-14

如图 7-15 所示，服务器发送 TCP 确认。

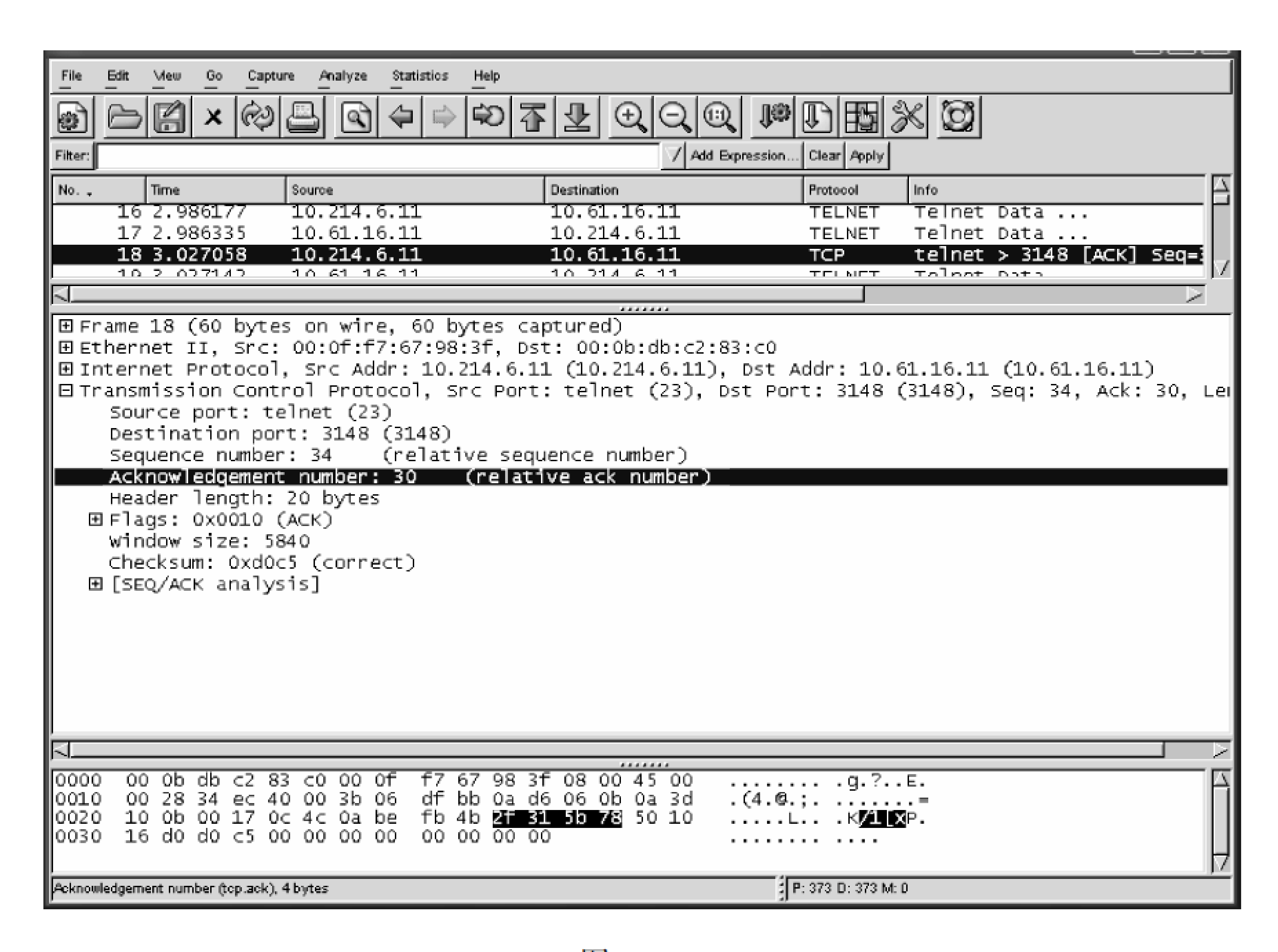



图 7-15

如图 7-16 所示，客户端发送关于协商窗口的尺寸的子协商。拒绝发送状态信息，同意远程流控制。

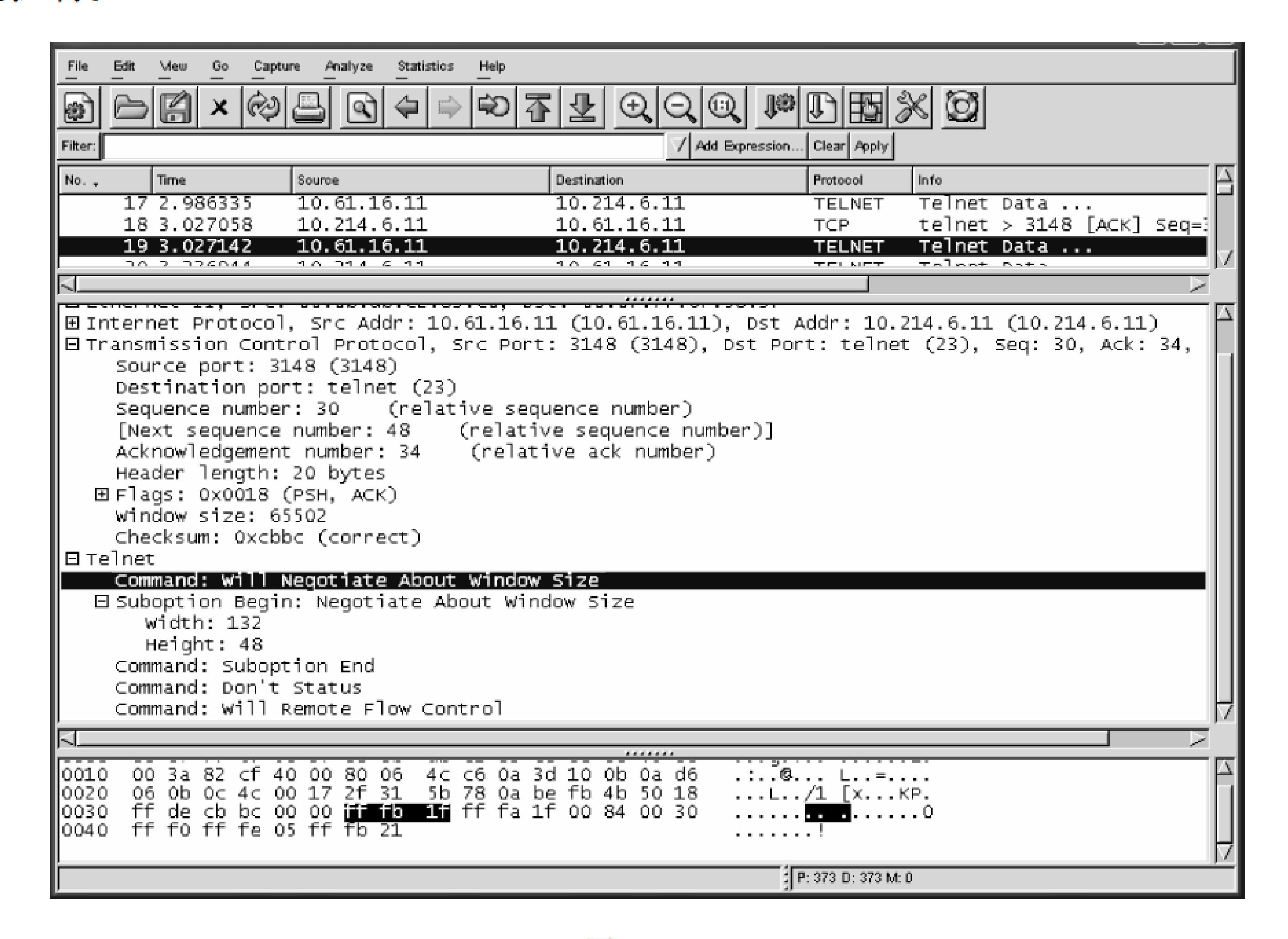


图　7-16

如图 7-17 所示，服务器将回显接收到的字符，同时服务器发送数据给客户端。

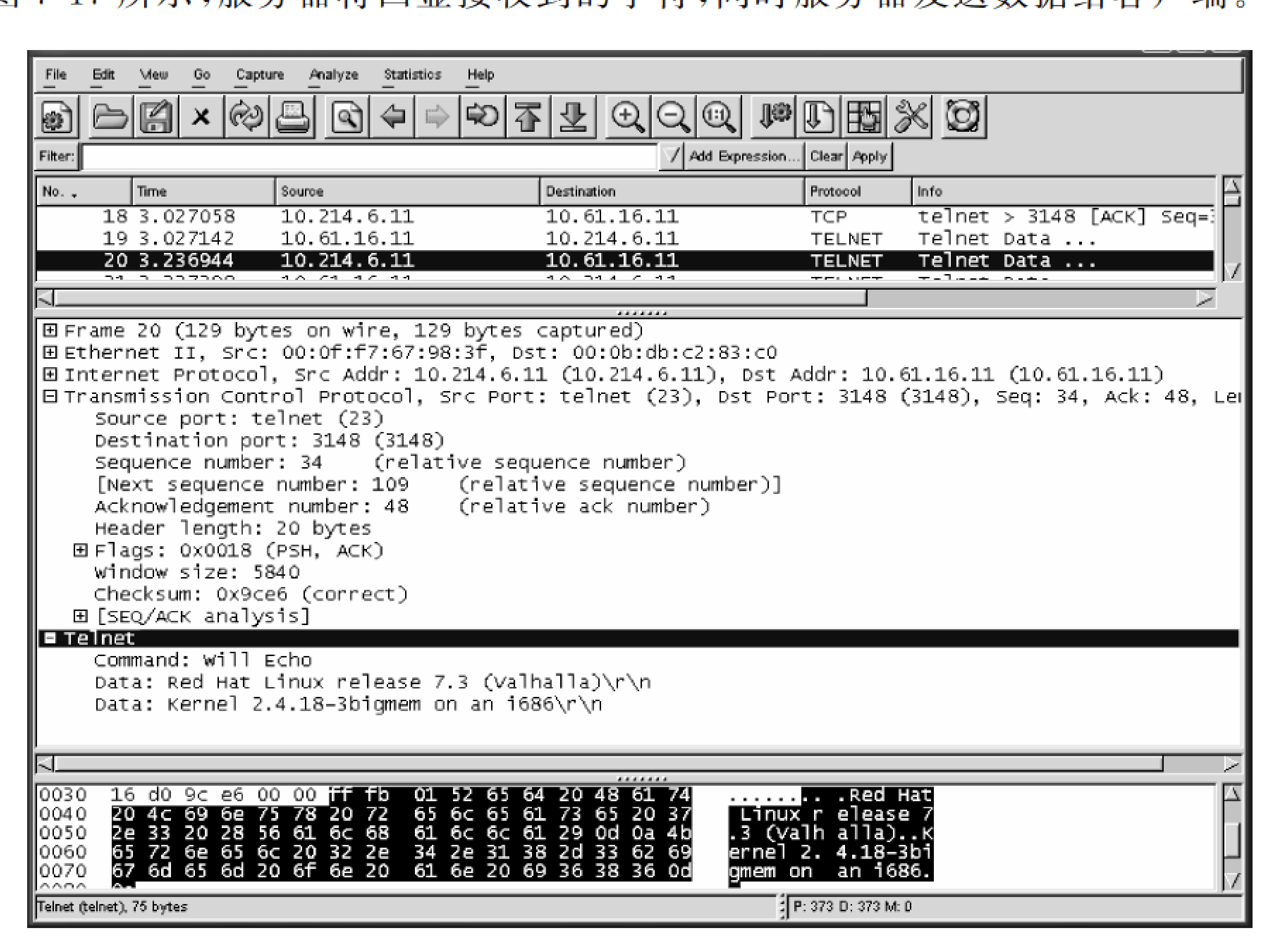


图　7-17

如图 7-18 所示，客户端同意服务器端回显接收到的字符。

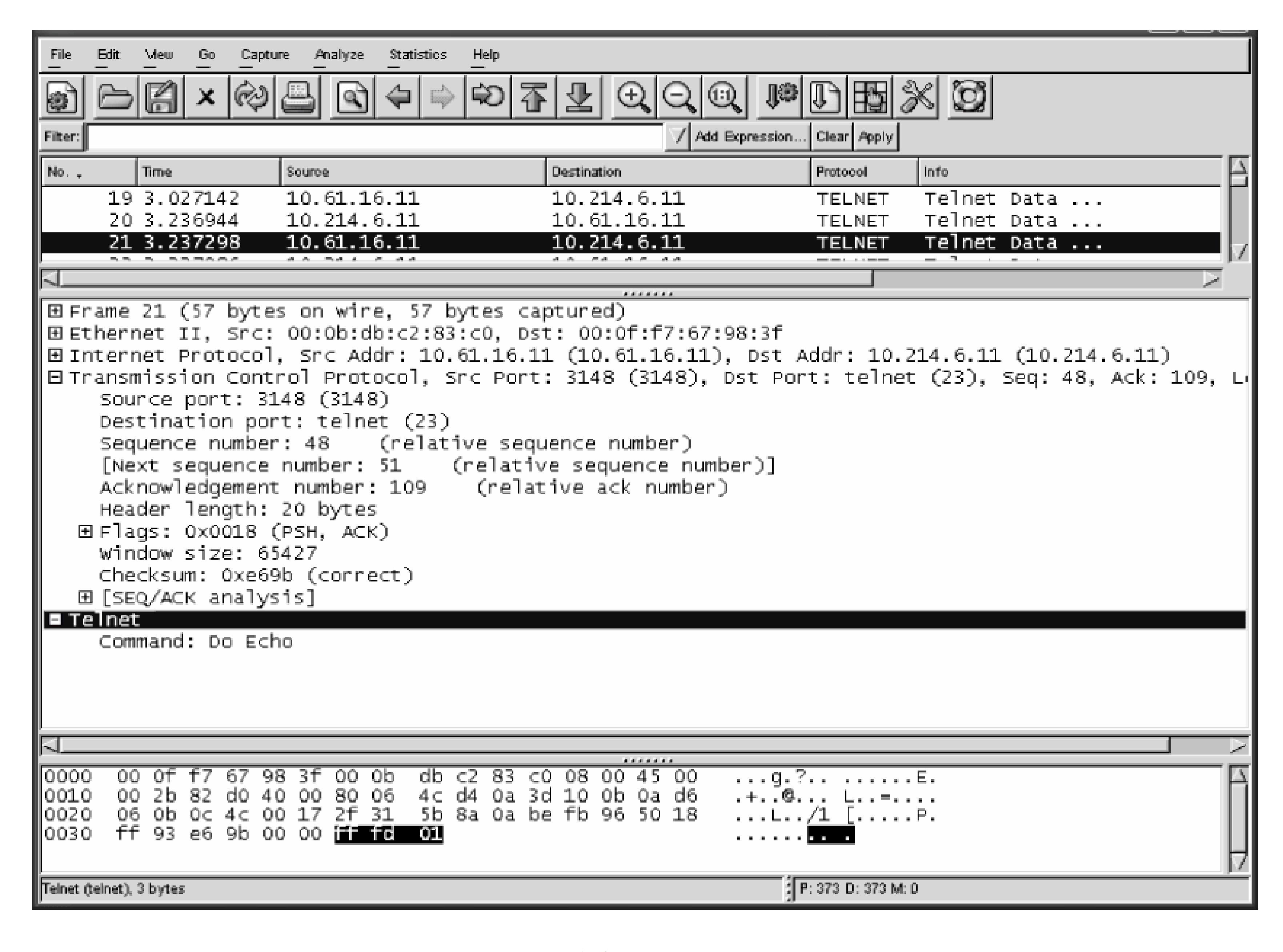



图 7-18

如图 7-19 所示，服务器端发送“login:”字符串。

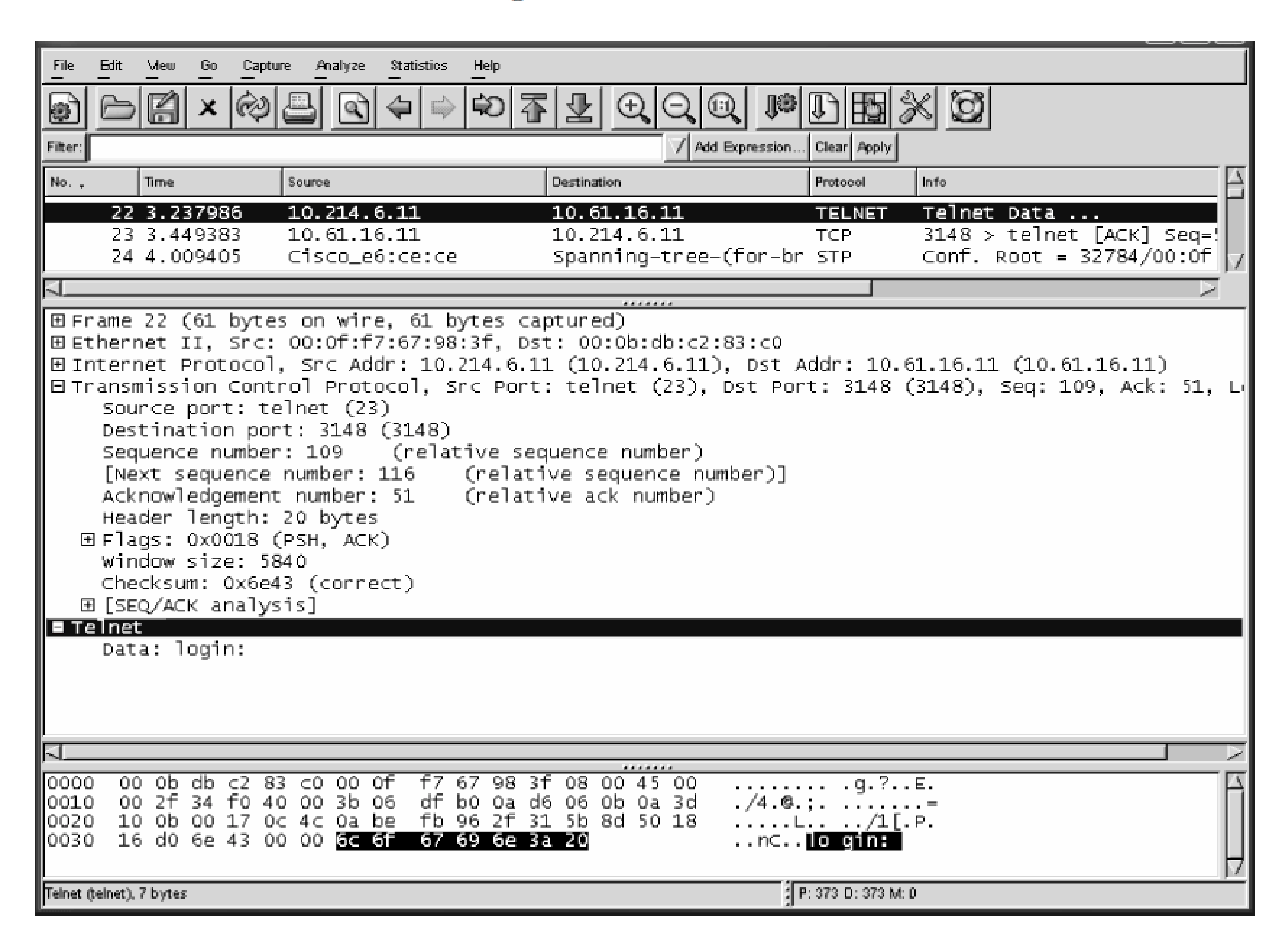



图 7-19

如图 7-20 所示，客户端发送 TCP 确认，确认收到字符串。

如图 7-21 所示，客户端发送用户名 root 中的第一个字符“r”。

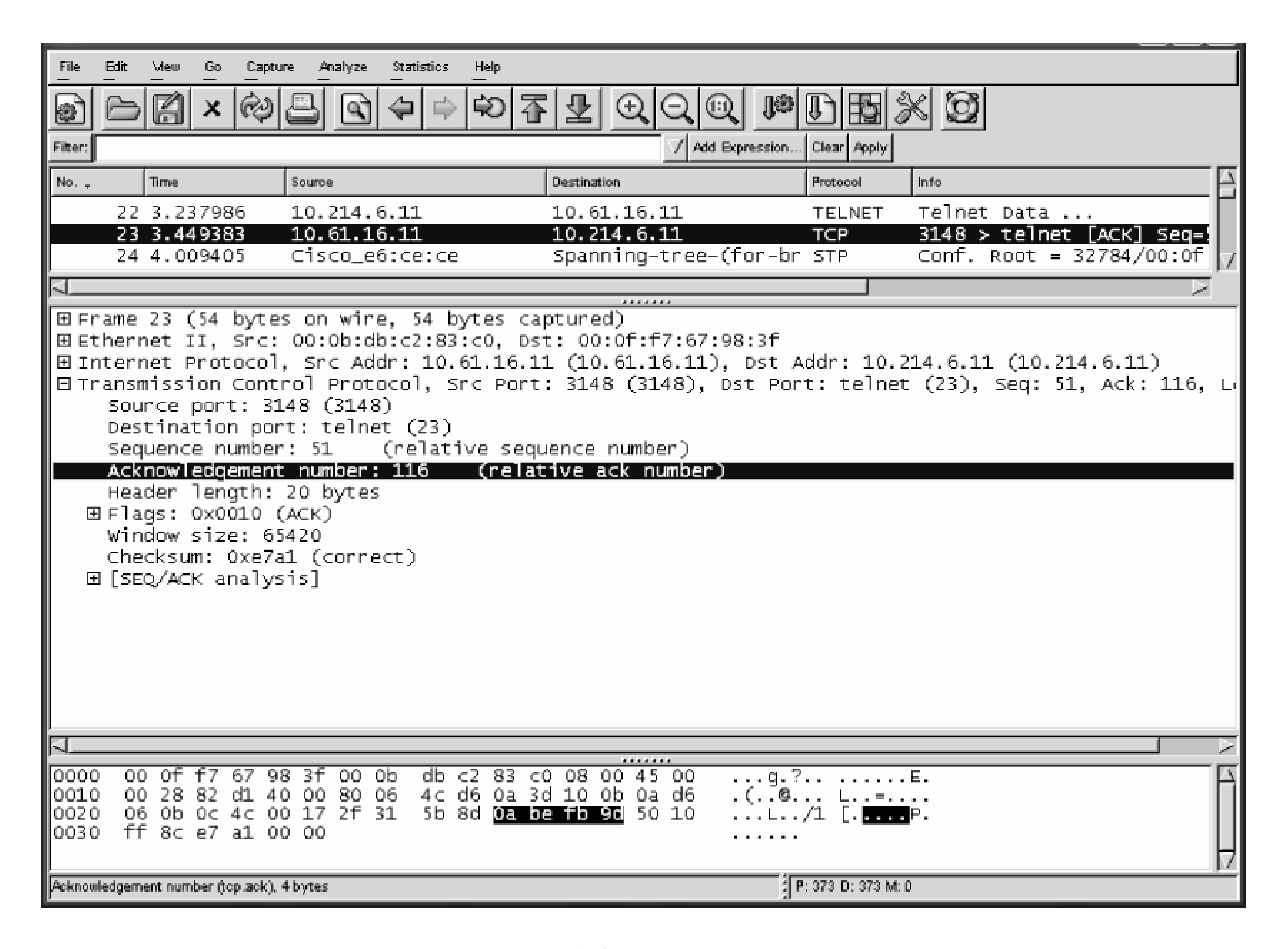



图 7-20

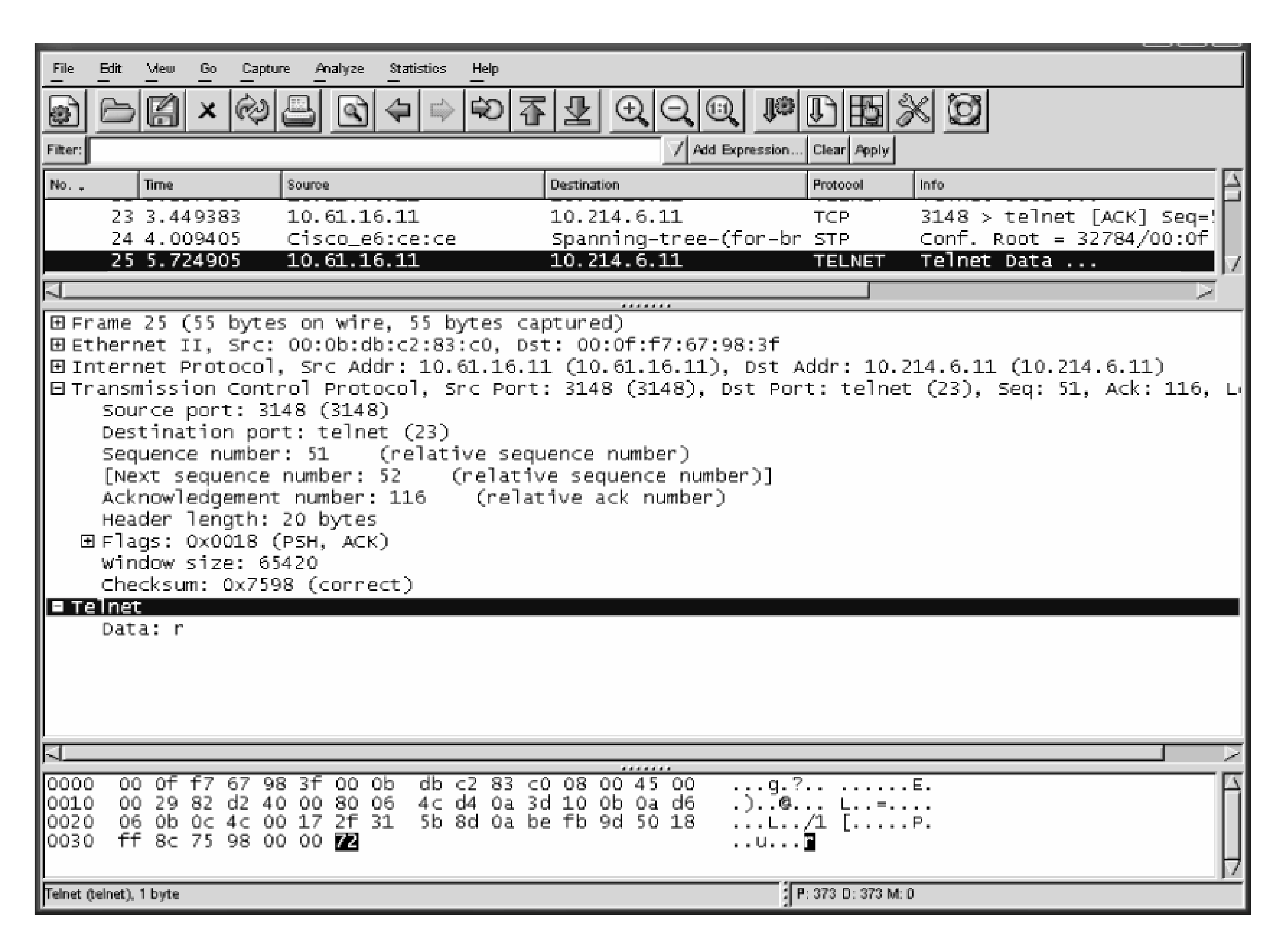



图 7-21

如图 7-22 所示，服务器端回显“r”。

如图 7-23 所示，客户端发送 TCP 确认，确认收到服务器端发送的回显。

图 7-21～图 7-23 是客户端输入“r”到回显“r”的一个完整的过程，接下是“o”“o”“t”字符的输入与回显，过程是一样的，在这里不再详细说明。

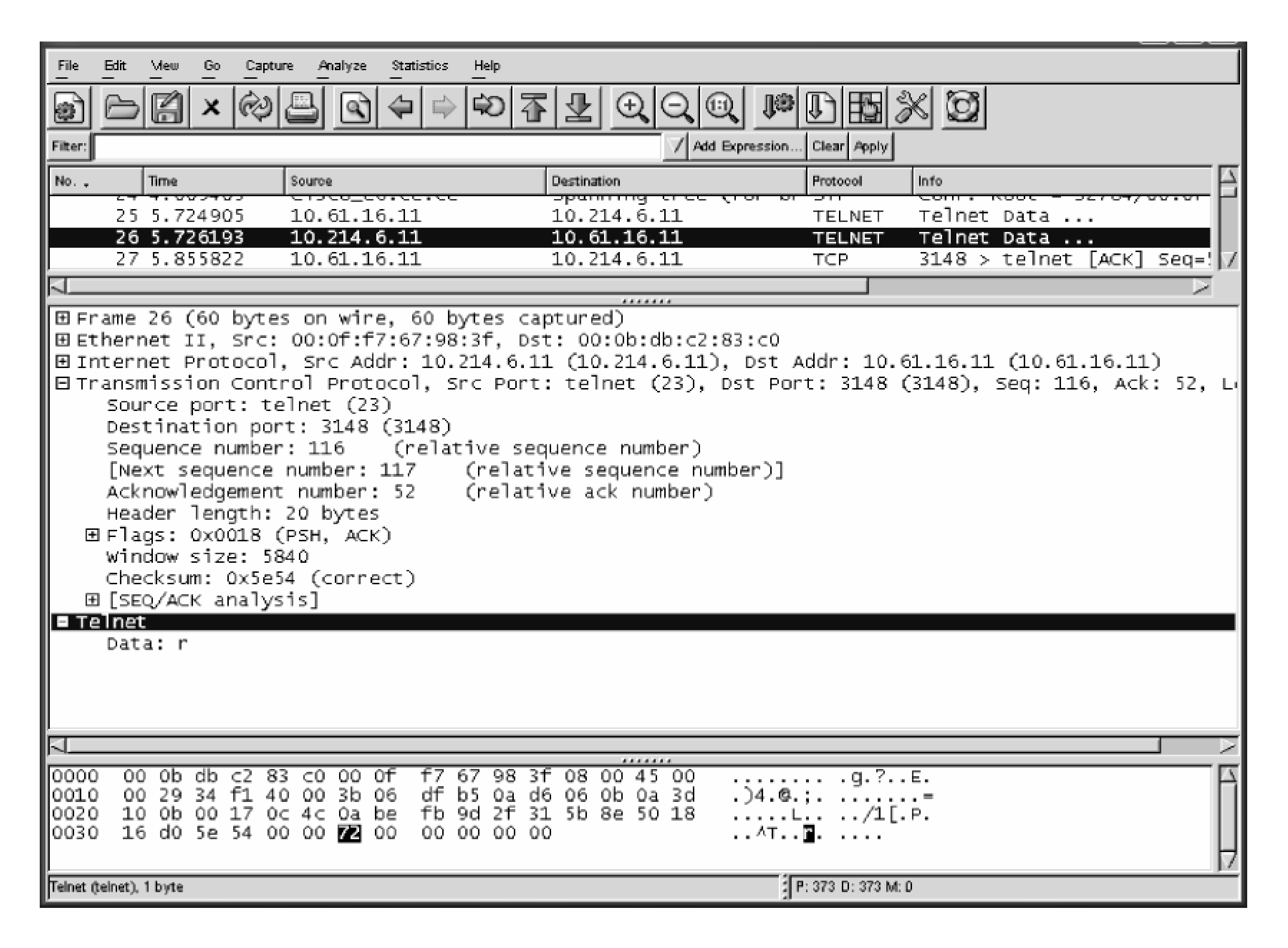



图 7-22

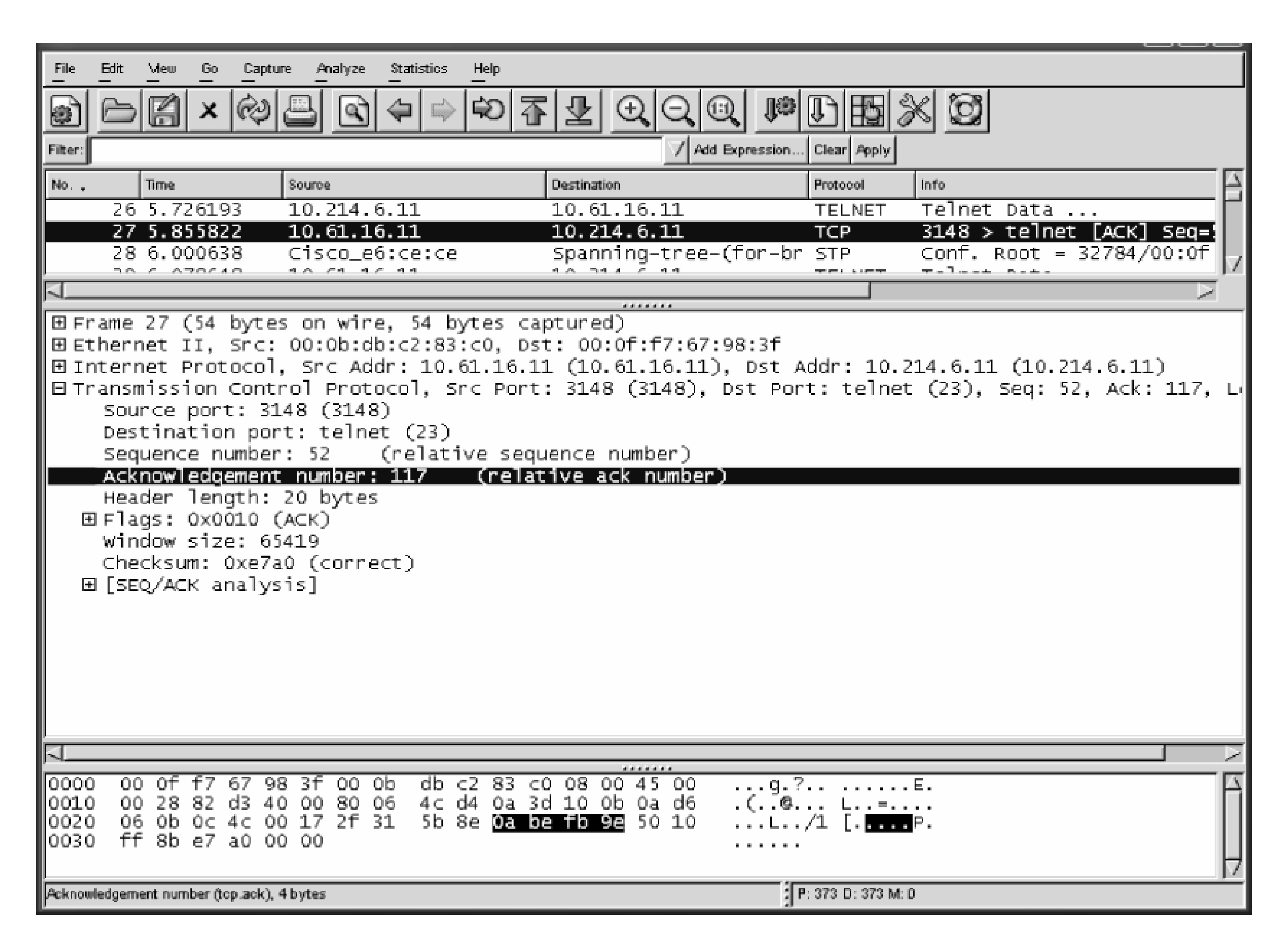



图 7-23

输完用户名,接下来如图 7-24 所示,服务器发送字符串“password:”。

如图 7-25 所示,客户端发送 TCP 确认。

如图 7-26 所示,客户端发送密码,可以看到图中密码的第一个字符为“c”。

如图 7-27 所示,服务器端发送 TCP 确认,确认收到“c”。

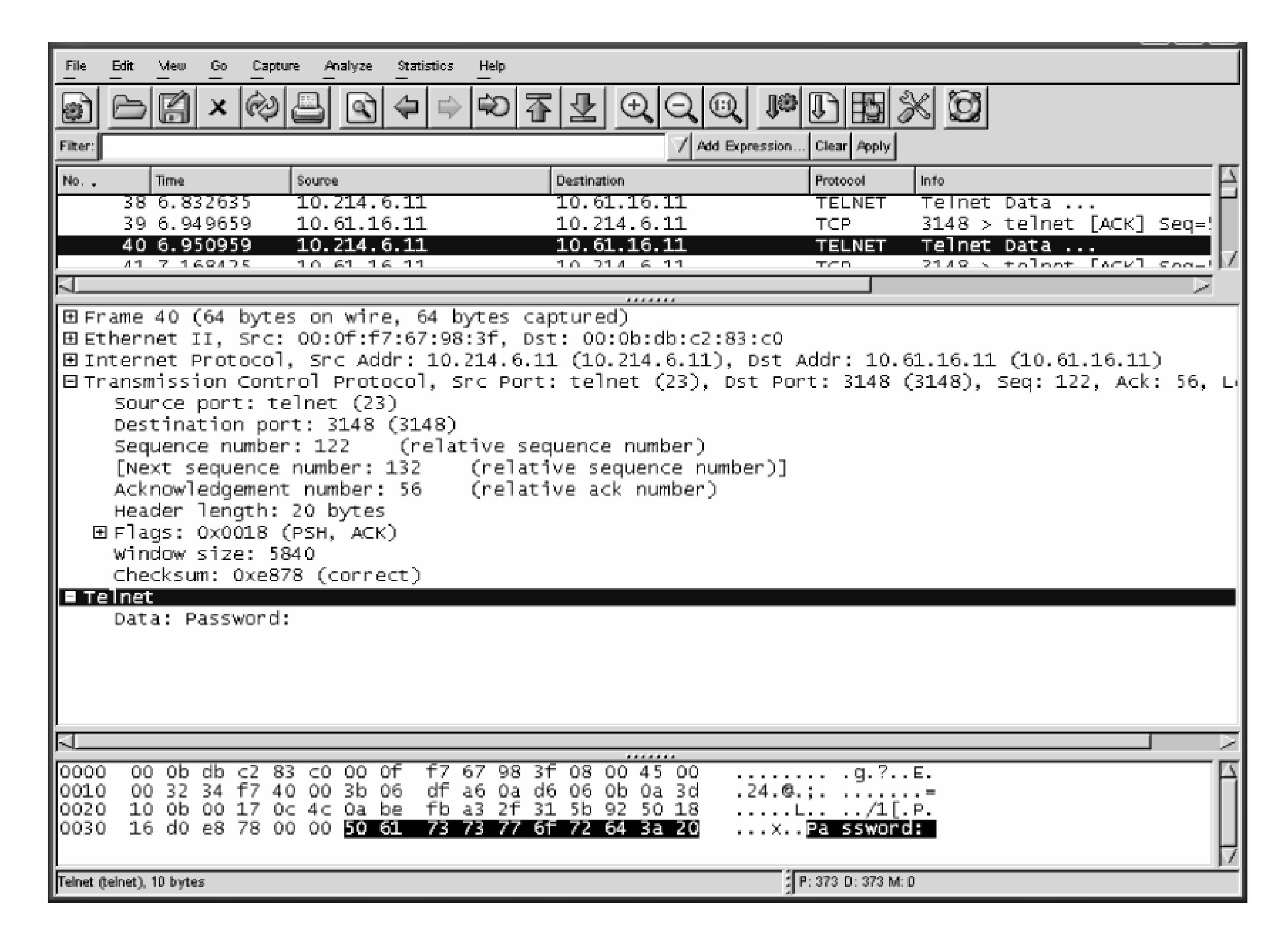

File Edit View Go Capture Analyze Statistics Help
Filter:
Add Expression... Clear Apply
No. . Time Source Destination Protocol Info
38 6.832635 10.214.6.11 10.61.16.11 TELNET Telnet Data ...
39 6.949659 10.61.16.11 10.214.6.11 TCP 3148 > telnet [ACK] Seq=
40 6.950959 10.214.6.11 10.61.16.11 TELNET Telnet Data ...
Frame 40 (64 bytes on wire, 64 bytes captured)
Ethernet II, Src: 00:0f:f7:67:98:3f, Dst: 00:0b:db:c2:83:c0
Internet Protocol, Src Addr: 10.214.6.11 (10.214.6.11), Dst Addr: 10.61.16.11 (10.61.16.11)
Transmission Control Protocol, Src Port: telnet (23), Dst Port: 3148 (3148), Seq: 122, Ack: 56, L
Source port: telnet (23)
Destination port: 3148 (3148)
Sequence number: 122 (relative sequence number)
[Next sequence number: 132 (relative sequence number)]
Acknowledgement number: 56 (relative ack number)
Header length: 20 bytes
Flags: 0x0018 (PSH, ACK)
Window size: 5840
Checksum: 0xe878 (correct)
Telnet
Data: Password:
0000 00 0b db c2 83 c0 00 0f f7 67 98 3f 08 00 45 00 ........ .g.?..E.
0010 00 32 34 f7 40 00 3b 06 df a6 0a d6 06 0b 0a 3d .24.@.;. .......=
0020 10 0b 00 17 0c 4c 0a be fb a3 2f 31 5b 92 50 18 .....L.. ../1[.P.
0030 16 d0 e8 78 00 00 50 61 73 73 77 6f 72 64 3a 20 ...x..Pa ssword:
Telnet (telnet), 10 bytes
P: 373 D: 373 M: 0

图 7-24

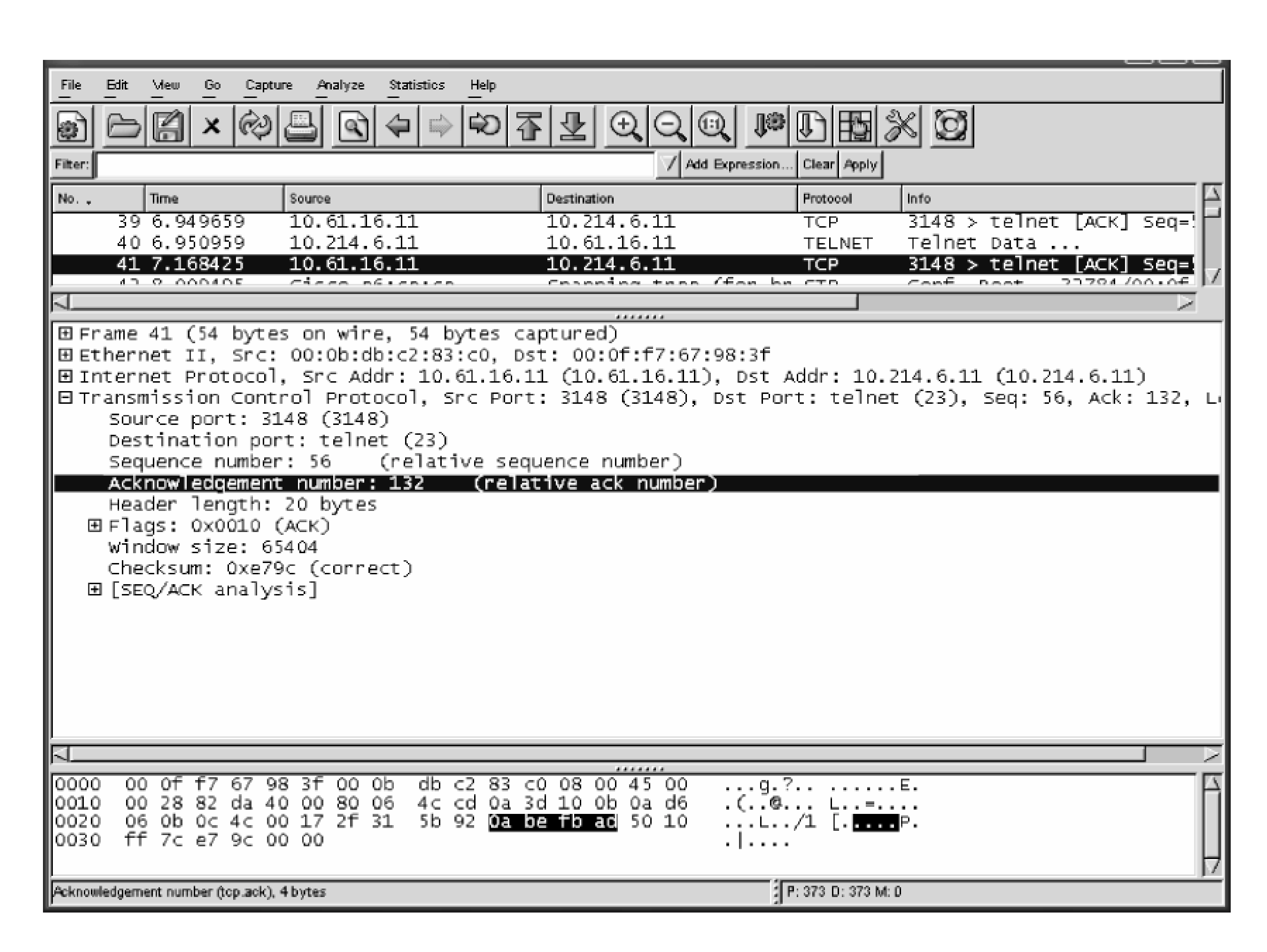

File Edit View Go Capture Analyze Statistics Help
Filter:
Add Expression... Clear Apply
No. . Time Source Destination Protocol Info
39 6.949659 10.61.16.11 10.214.6.11 TCP 3148 > telnet [ACK] Seq=
40 6.950959 10.214.6.11 10.61.16.11 TELNET Telnet Data ...
41 7.168425 10.61.16.11 10.214.6.11 TCP 3148 > telnet [ACK] Seq=
Frame 41 (54 bytes on wire, 54 bytes captured)
Ethernet II, Src: 00:0b:db:c2:83:c0, Dst: 00:0f:f7:67:98:3f
Internet Protocol, Src Addr: 10.61.16.11 (10.61.16.11), Dst Addr: 10.214.6.11 (10.214.6.11)
Transmission Control Protocol, Src Port: 3148 (3148), Dst Port: telnet (23), Seq: 56, Ack: 132, L
Source port: 3148 (3148)
Destination port: telnet (23)
Sequence number: 56 (relative sequence number)
Acknowledgement number: 132 (relative ack number)
Header length: 20 bytes
Flags: 0x0010 (ACK)
Window size: 65404
Checksum: 0xe79c (correct)
[SEQ/ACK analysis]
0000 00 0f f7 67 98 3f 00 0b db c2 83 c0 08 00 45 00 ...g.?.. ......E.
0010 00 28 82 da 40 00 80 06 4c cd 0a 3d 10 0b 0a d6 .(..@... L..=....
0020 06 0b 0c 4c 00 17 2f 31 5b 92 0a be fb ad 50 10 ...L../1 [.....P.
0030 ff 7c e7 9c 00 00 .|....
Acknowledgement number (tcp.ack), 4 bytes
P: 373 D: 373 M: 0

图 7-25

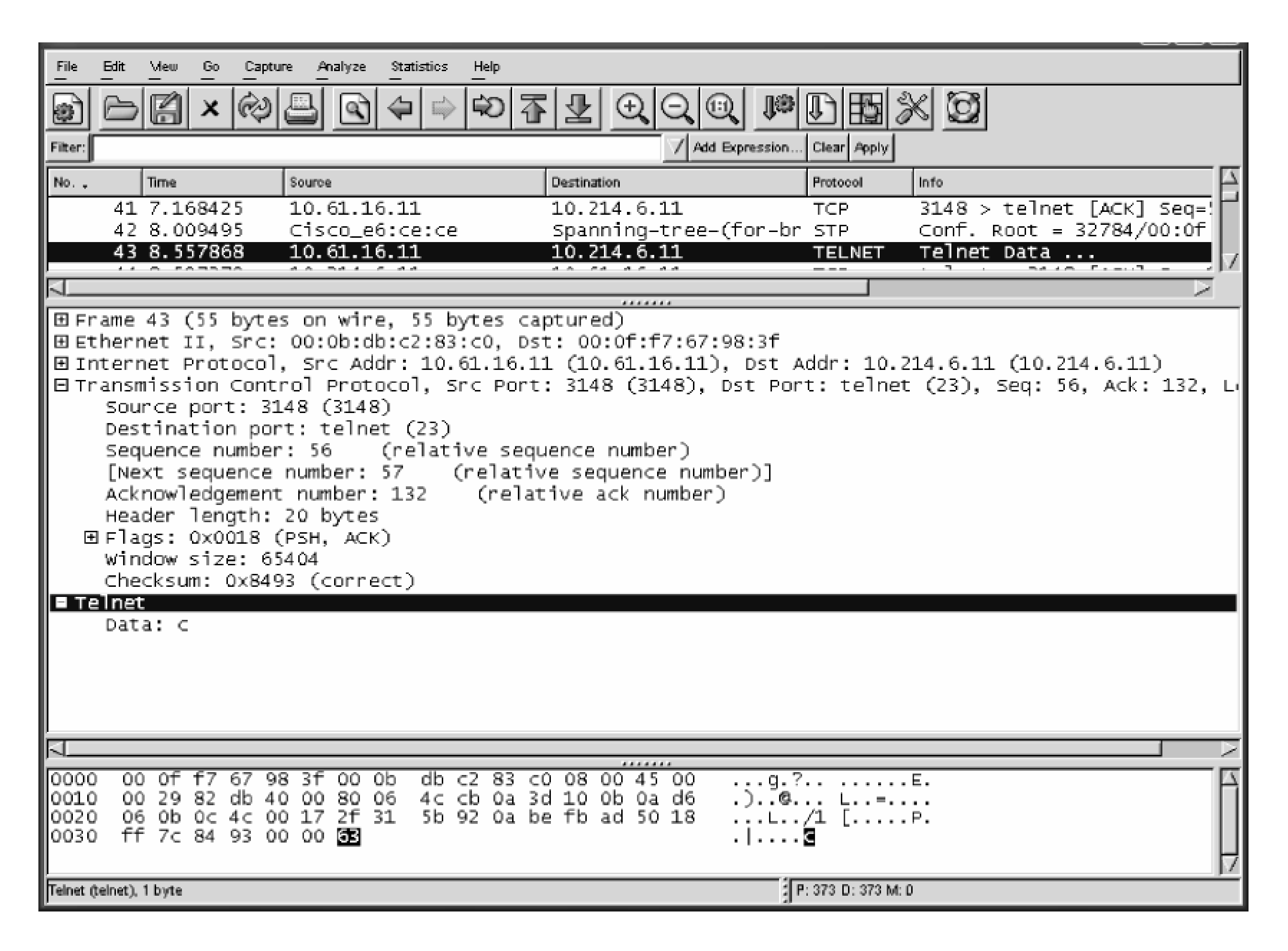

File Edit View Go Capture Analyze Statistics Help
Filter: Add Expression... Clear Apply
No. Time Source Destination Protocol Info
41 7.168425 10.61.16.11 10.214.6.11 TCP 3148 > telnet [ACK] Seq=
42 8.009495 Cisco_e6:ce:ce Spanning-tree-(for-br STP Conf. Root = 32784/00:0f
43 8.557868 10.61.16.11 10.214.6.11 TELNET Telnet Data ...
Frame 43 (55 bytes on wire, 55 bytes captured)
Ethernet II, Src: 00:0b:db:c2:83:c0, Dst: 00:0f:f7:67:98:3f
Internet Protocol, Src Addr: 10.61.16.11 (10.61.16.11), Dst Addr: 10.214.6.11 (10.214.6.11)
Transmission Control Protocol, Src Port: 3148 (3148), Dst Port: telnet (23), Seq: 56, Ack: 132, L
Source port: 3148 (3148)
Destination port: telnet (23)
Sequence number: 56 (relative sequence number)
[Next sequence number: 57 (relative sequence number)]
Acknowledgement number: 132 (relative ack number)
Header length: 20 bytes
Flags: 0x0018 (PSH, ACK)
Window size: 65404
Checksum: 0x8493 (correct)
Telnet
Data: c
0000 00 0f f7 67 98 3f 00 0b db c2 83 c0 08 00 45 00 ...g.?.. ......E.
0010 00 29 82 db 40 00 80 06 4c cb 0a 3d 10 0b 0a d6 .)..@... L..=....
0020 06 0b 0c 4c 00 17 2f 31 5b 92 0a be fb ad 50 18 ...L../1 [.....P.
0030 ff 7c 84 93 00 00 63 .|....c
Telnet (telnet), 1 byte
P: 373 D: 373 M: 0

图 7-26

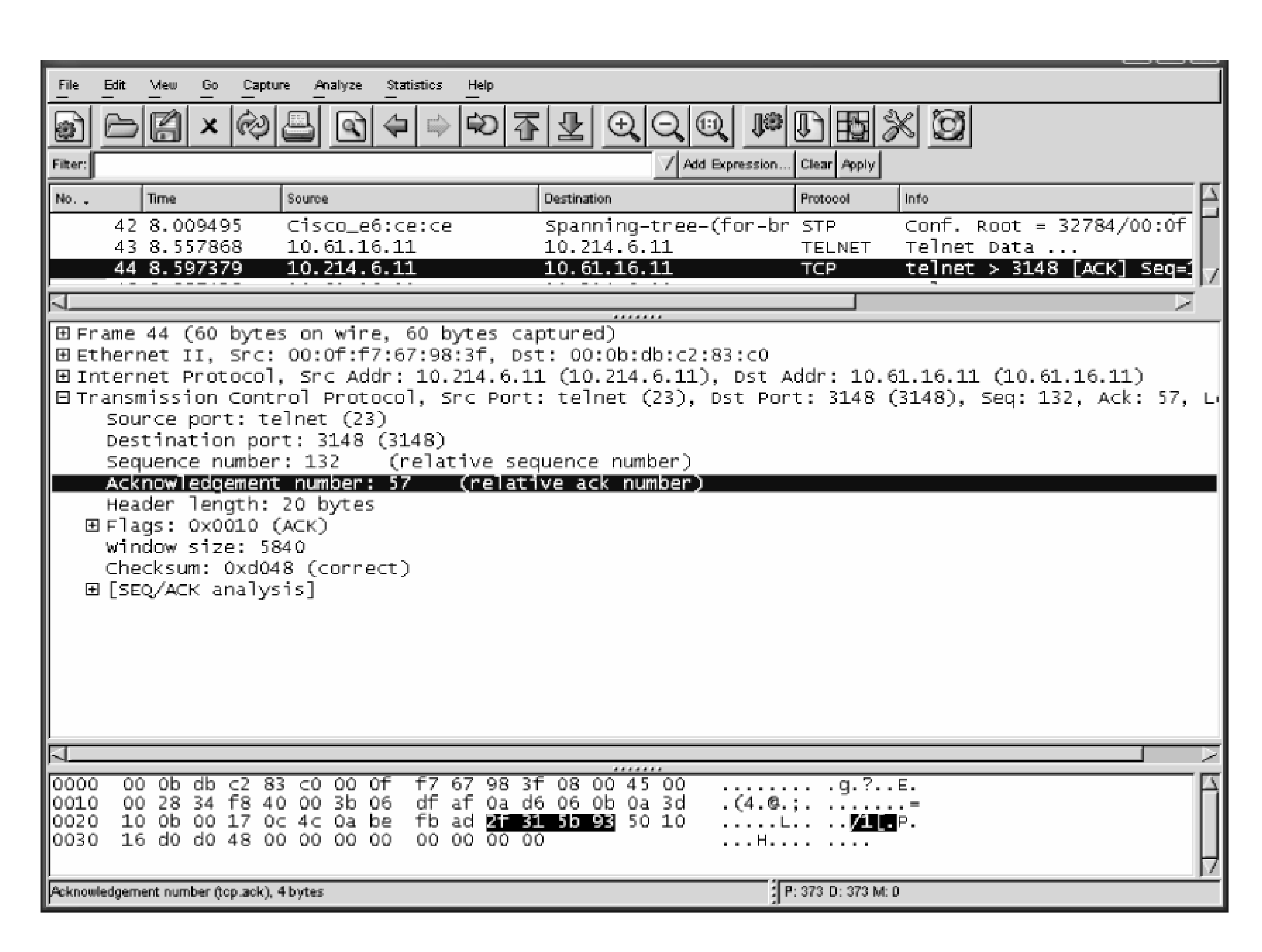

File Edit View Go Capture Analyze Statistics Help
Filter: Add Expression... Clear Apply
No. Time Source Destination Protocol Info
42 8.009495 Cisco_e6:ce:ce Spanning-tree-(for-br STP Conf. Root = 32784/00:0f
43 8.557868 10.61.16.11 10.214.6.11 TELNET Telnet Data ...
44 8.597379 10.214.6.11 10.61.16.11 TCP telnet > 3148 [ACK] Seq=
Frame 44 (60 bytes on wire, 60 bytes captured)
Ethernet II, Src: 00:0f:f7:67:98:3f, Dst: 00:0b:db:c2:83:c0
Internet Protocol, Src Addr: 10.214.6.11 (10.214.6.11), Dst Addr: 10.61.16.11 (10.61.16.11)
Transmission Control Protocol, Src Port: telnet (23), Dst Port: 3148 (3148), Seq: 132, Ack: 57, L
Source port: telnet (23)
Destination port: 3148 (3148)
Sequence number: 132 (relative sequence number)
Acknowledgement number: 57 (relative ack number)
Header length: 20 bytes
Flags: 0x0010 (ACK)
Window size: 5840
Checksum: 0xd048 (correct)
[SEQ/ACK analysis]
0000 00 0b db c2 83 c0 00 0f f7 67 98 3f 08 00 45 00 ........ .g.?..E.
0010 00 28 34 f8 40 00 3b 06 df af 0a d6 06 0b 0a 3d .(4.@.;. .......=
0020 10 0b 00 17 0c 4c 0a be fb ad 2f 31 5b 93 50 10 .....L.. ../1[.P.
0030 16 d0 d0 48 00 00 00 00 00 00 00 00 ...H.... ....
Acknowledgement number (tcp.ack), 4 bytes
P: 373 D: 373 M: 0

图 7-27

如图 7-28 所示，客户端发送密码的第二个字符“g”：

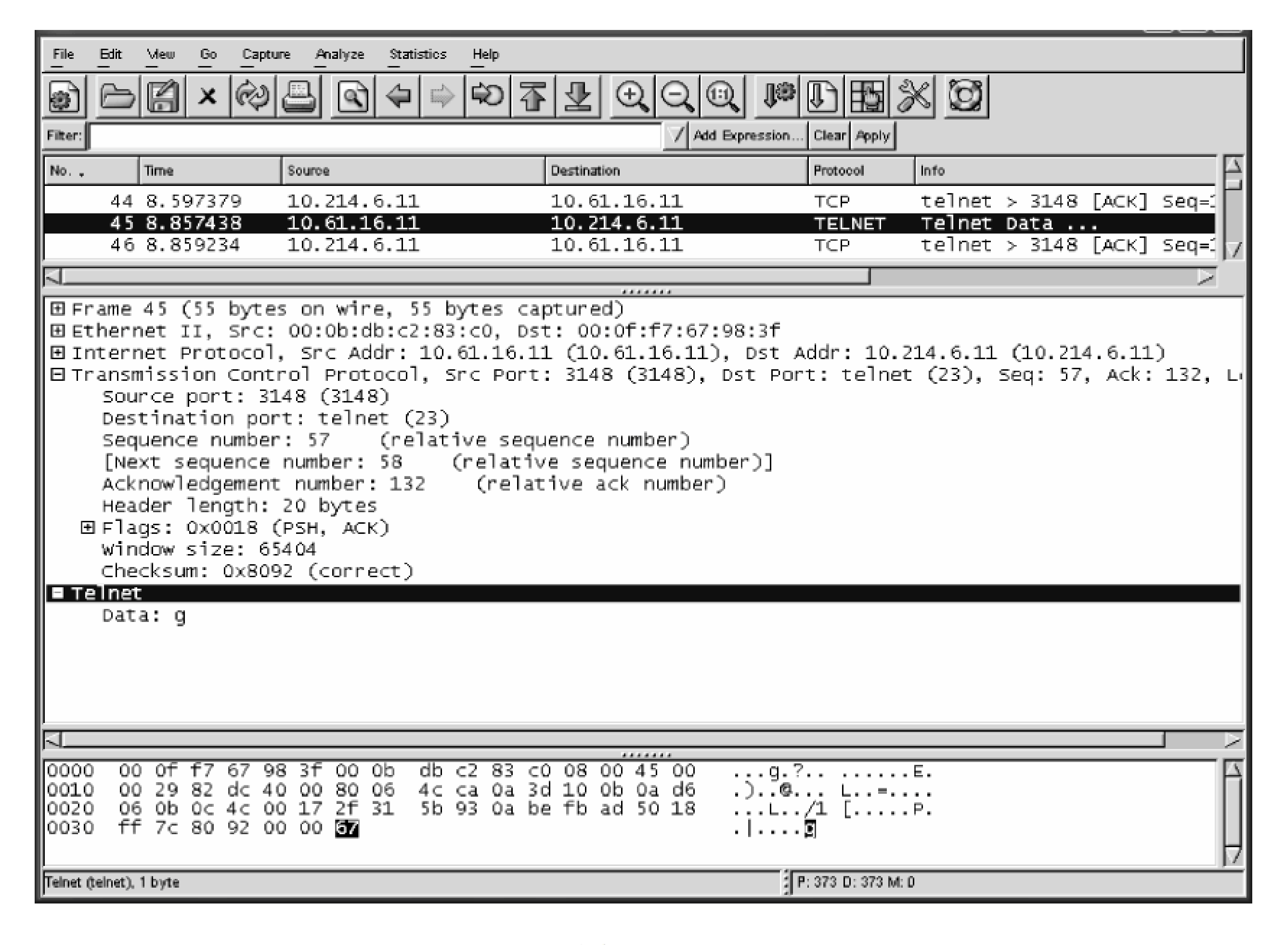



图　7-28

如图 7-29 所示，服务器端发送 TCP 确认，确认收到字符“g”。

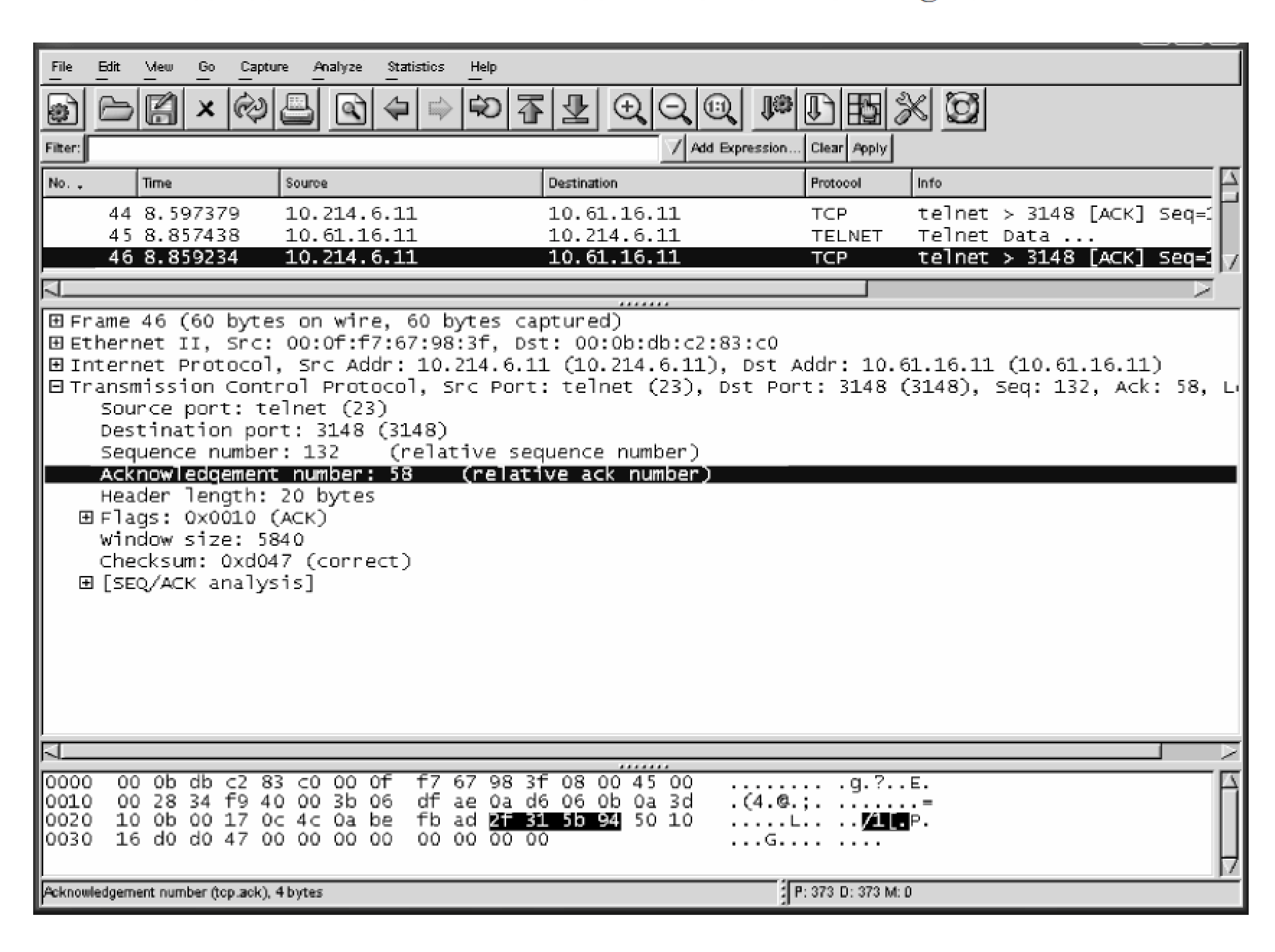



图　7-29

就这样，客户端从键盘输入密码，并一个字符接一个字符发送给服务器端，服务器端对每个收到的字符进行 TCP 确认，直到密码输入完毕，这个过程和前面 login 过程不同，服务

器端不回显 password 部分，只发送 TCP 确认给客户端。

Telnet 用户成功登录后，服务器端发送数据给客户端，如图 7-30 所示，服务器发送的数据，Telnet 协议定义回车换行(CR-LF)序列来代表行结束，图 7-30 中用\r\n 表示，客户端收到数据后发送 TCP 确认如图 7-31 所示，接着服务器端继续发送数据，如图 7-32 所示。

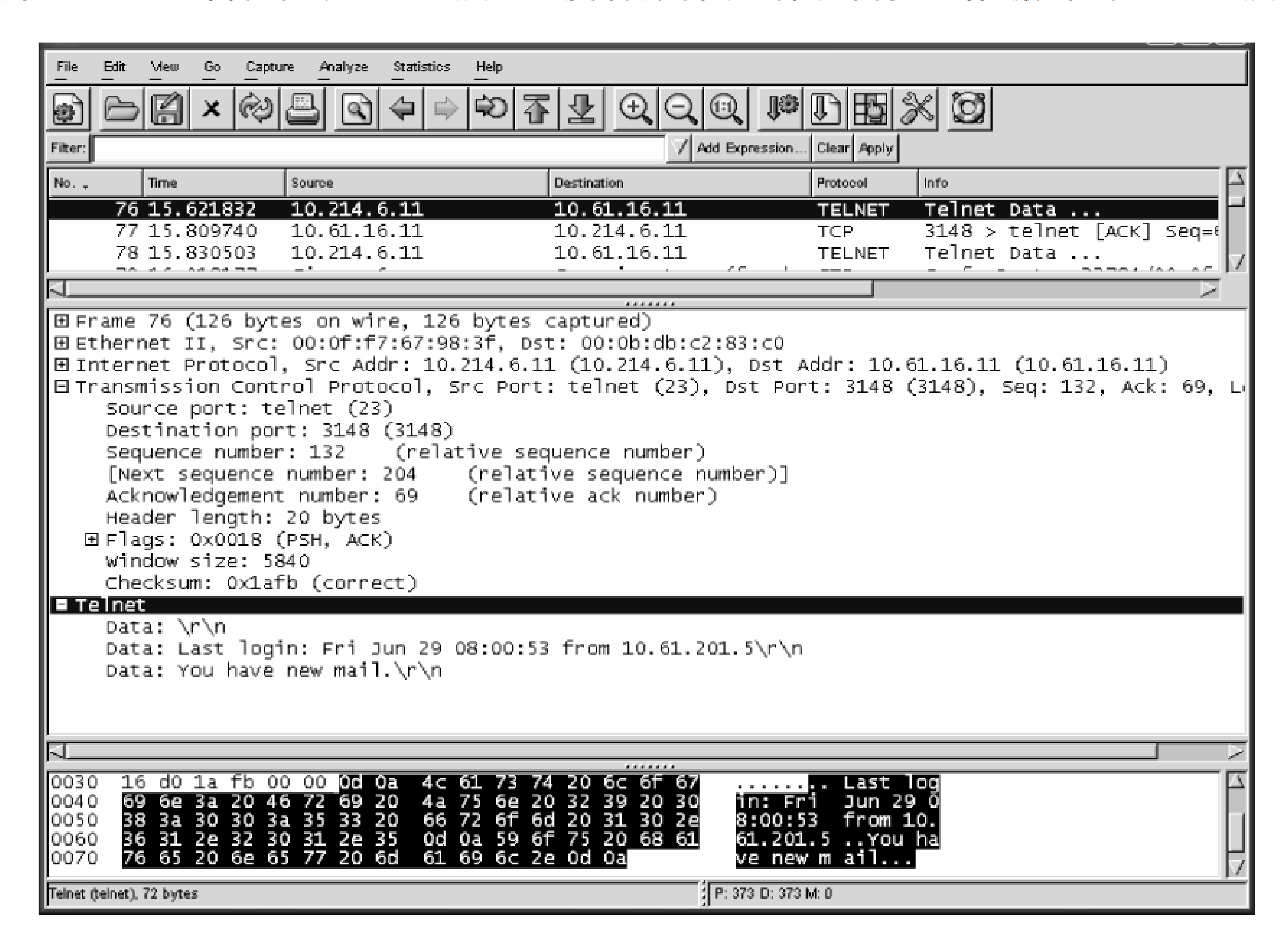



图 7-30

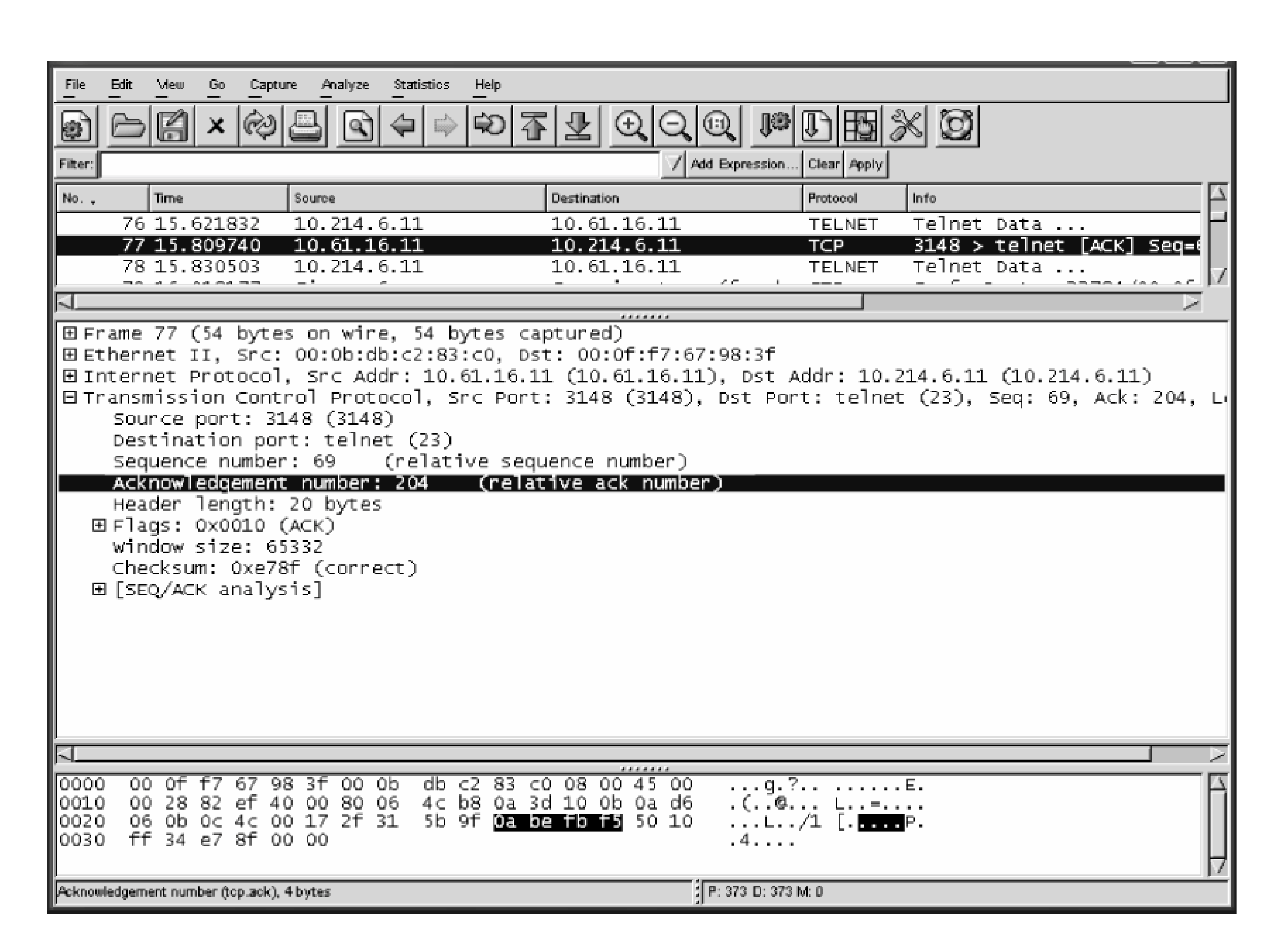



图 7-31

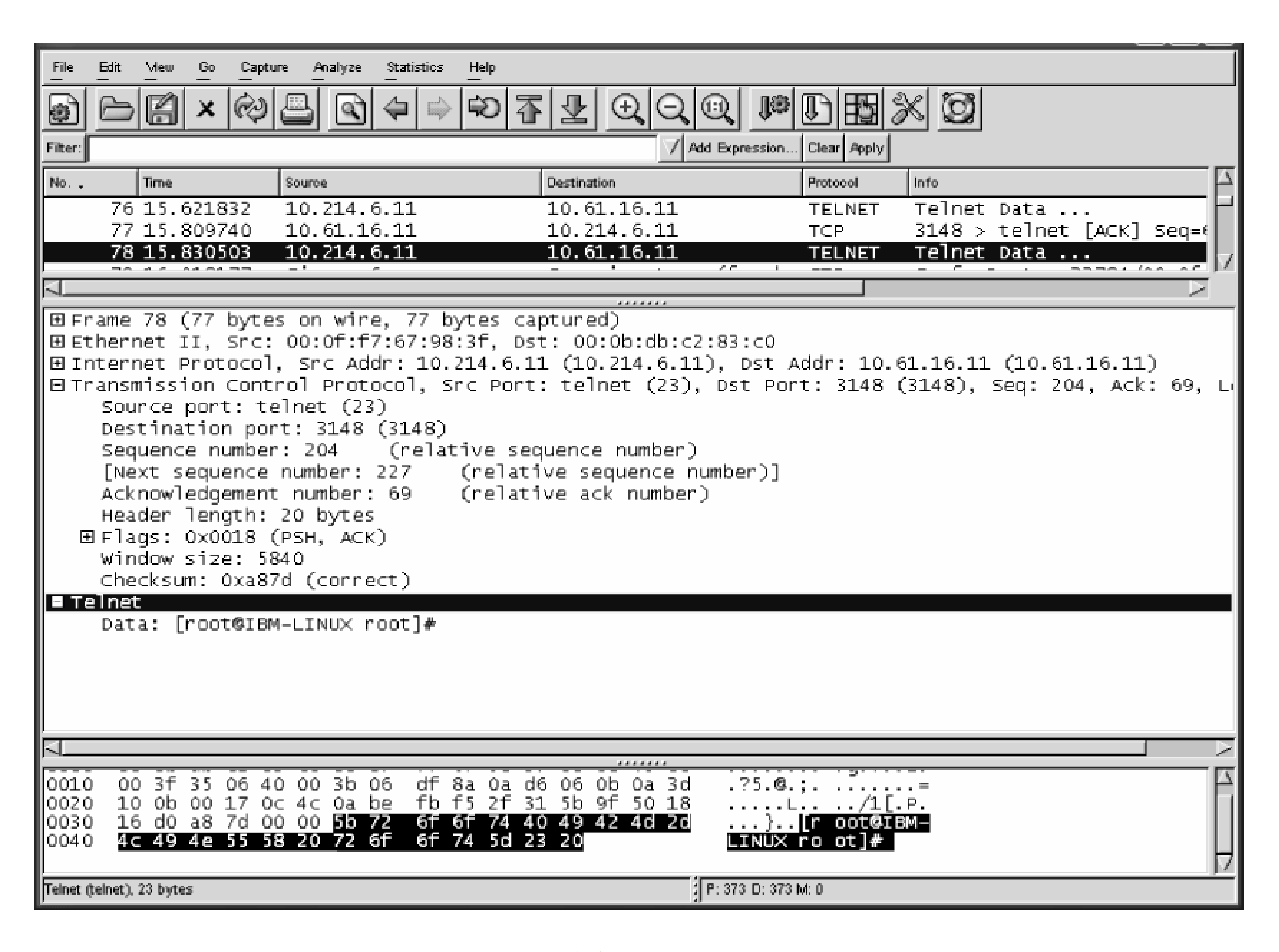



图　7-32

在键盘上输入 ls 命令，来看看客户端和服务器直接发送的数据，如图 7-33 所示，客户端发送数据“l”。

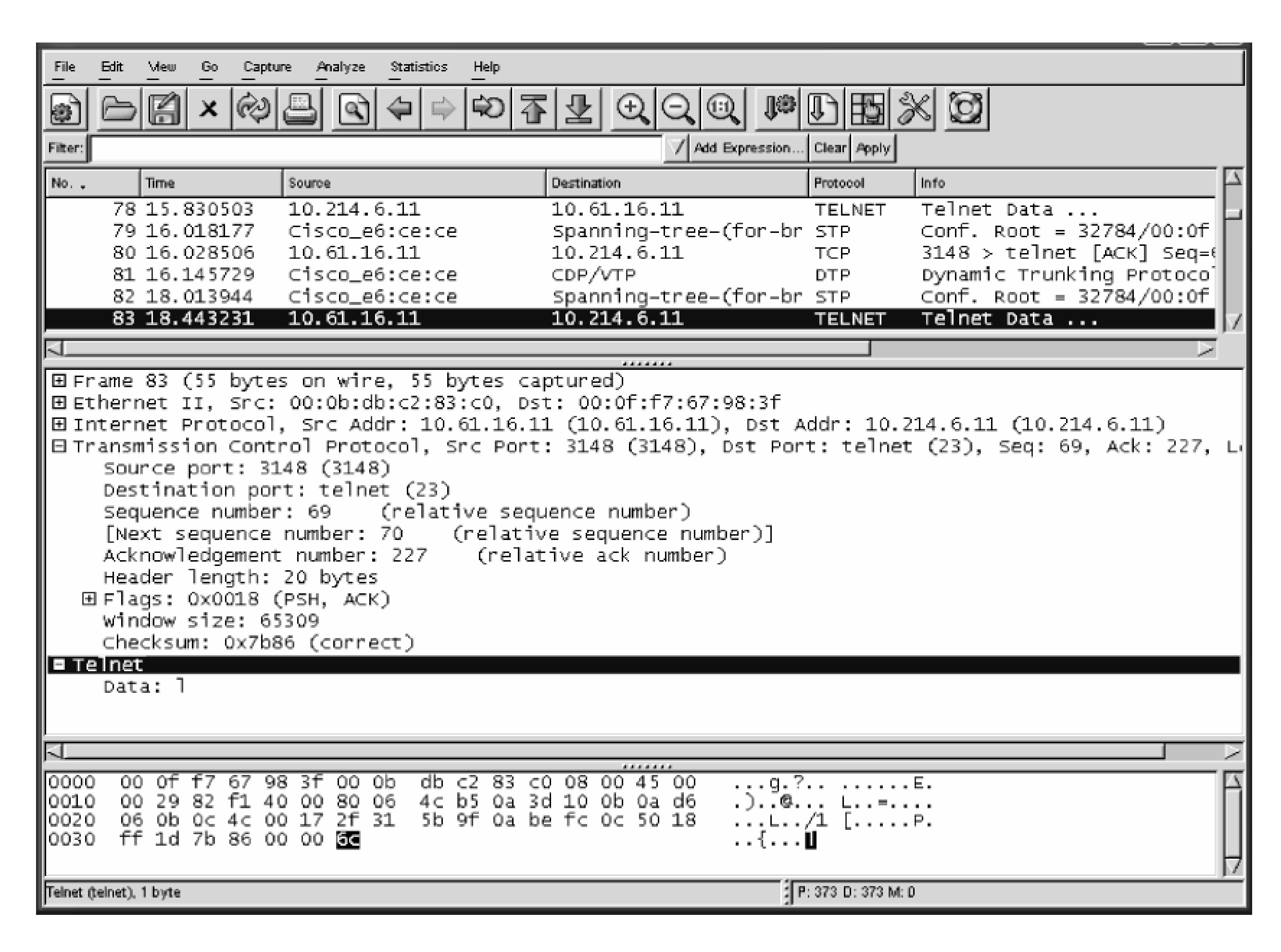



图　7-33

如图 7-34 所示，服务器端回显“l”。

如图 7-35 所示，客户端发送数据“s”。

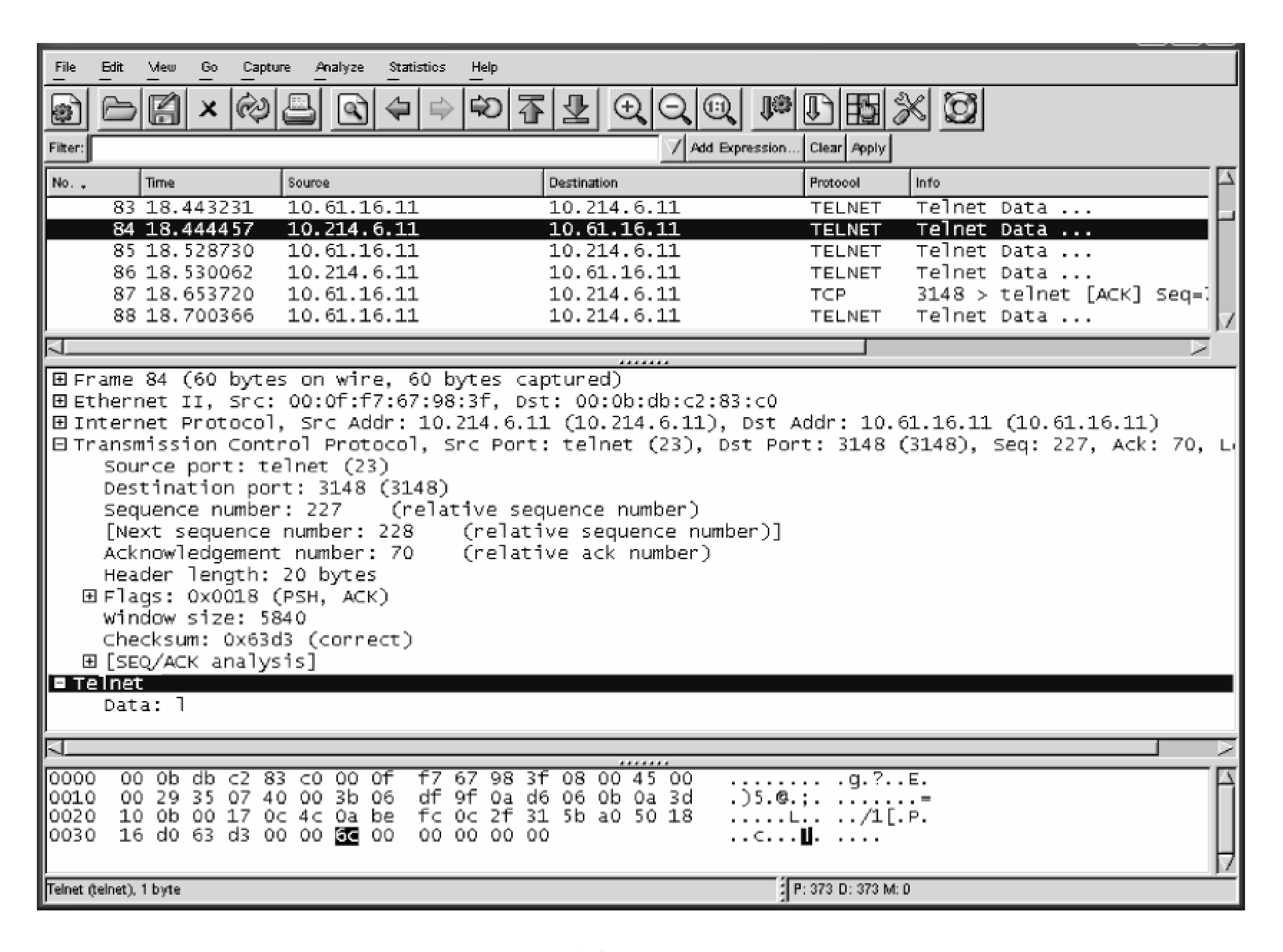



图 7-34

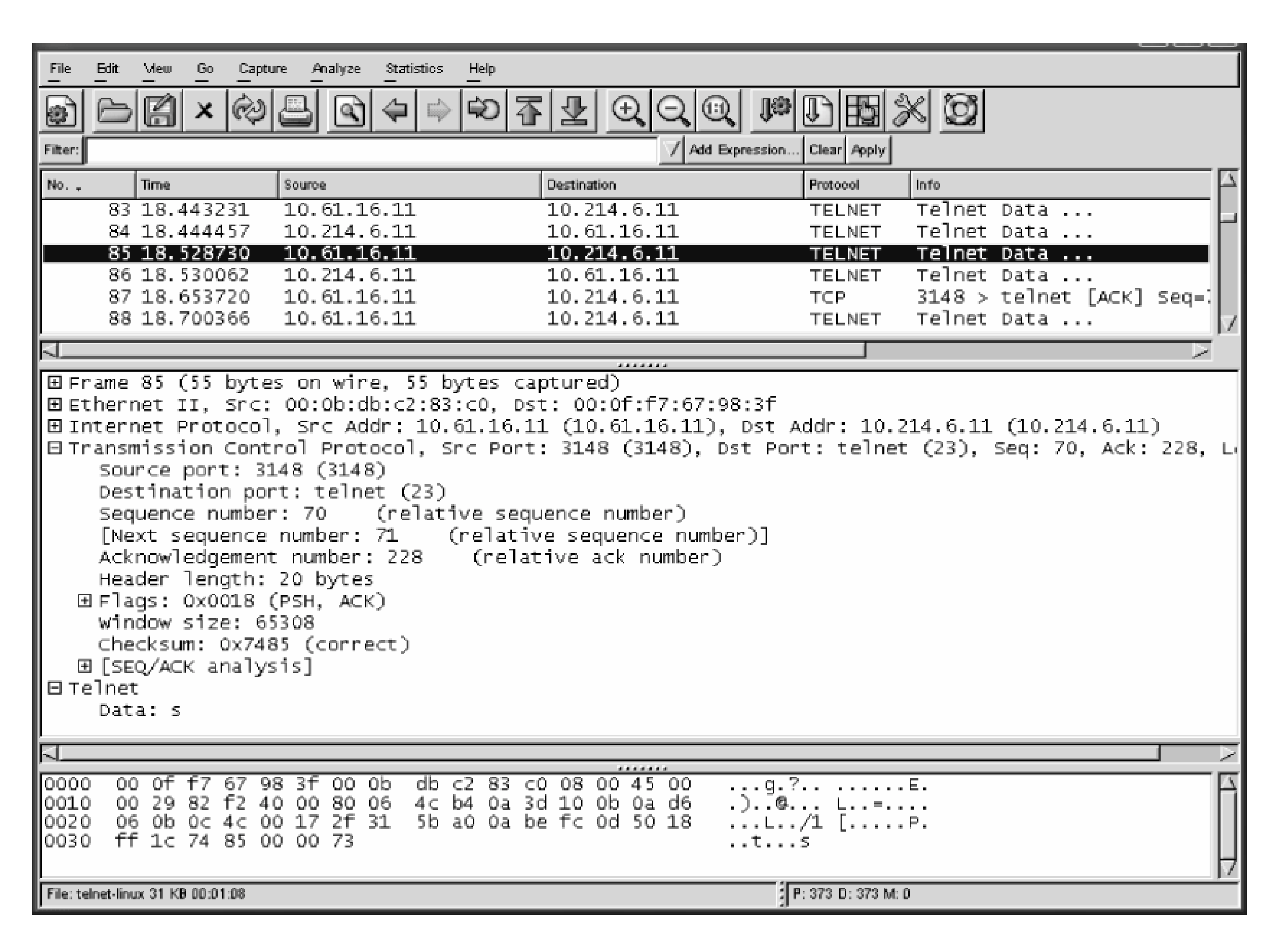



图 7-35

如图 7-36 所示，服务器端回显“s”。

如图 7-37 所示，客户端发送 TCP 确认。

如图 7-38 所示，在键盘上输入回车时，客户端就会发送数据“\r”。

如图 7-39 所示，服务器端回显换行命令。

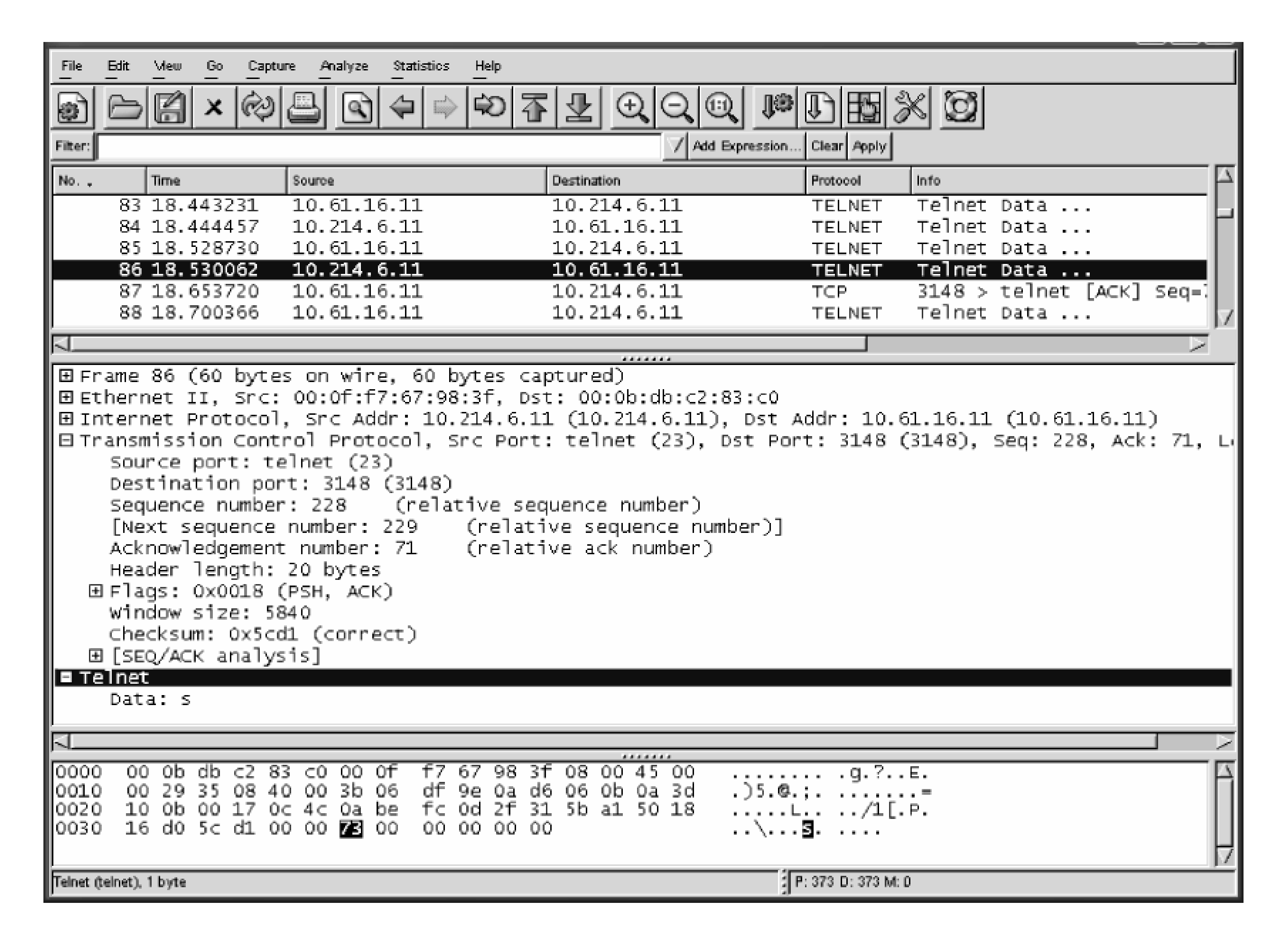

File Edit View Go Capture Analyze Statistics Help
Filter: Add Expression... Clear Apply
No. . Time Source Destination Protocol Info
83 18.443231 10.61.16.11 10.214.6.11 TELNET Telnet Data ...
84 18.444457 10.214.6.11 10.61.16.11 TELNET Telnet Data ...
85 18.528730 10.61.16.11 10.214.6.11 TELNET Telnet Data ...
86 18.530062 10.214.6.11 10.61.16.11 TELNET Telnet Data ...
87 18.653720 10.61.16.11 10.214.6.11 TCP 3148 > telnet [ACK] Seq=
88 18.700366 10.61.16.11 10.214.6.11 TELNET Telnet Data ...
⊞ Frame 86 (60 bytes on wire, 60 bytes captured)
⊞ Ethernet II, Src: 00:0f:f7:67:98:3f, Dst: 00:0b:db:c2:83:c0
⊞ Internet Protocol, Src Addr: 10.214.6.11 (10.214.6.11), Dst Addr: 10.61.16.11 (10.61.16.11)
⊟ Transmission Control Protocol, Src Port: telnet (23), Dst Port: 3148 (3148), Seq: 228, Ack: 71, L
Source port: telnet (23)
Destination port: 3148 (3148)
Sequence number: 228 (relative sequence number)
[Next sequence number: 229 (relative sequence number)]
Acknowledgement number: 71 (relative ack number)
Header length: 20 bytes
⊞ Flags: 0x0018 (PSH, ACK)
Window size: 5840
Checksum: 0x5cd1 (correct)
⊞ [SEQ/ACK analysis]
⊟ Telnet
Data: s
0000 00 0b db c2 83 c0 00 0f f7 67 98 3f 08 00 45 00 ........ .g.?..E.
0010 00 29 35 08 40 00 3b 06 df 9e 0a d6 06 0b 0a 3d .)5.@.;. .......=
0020 10 0b 00 17 0c 4c 0a be fc 0d 2f 31 5b a1 50 18 .....L.. ../1[.P.
0030 16 d0 5c d1 00 00 73 00 00 00 00 00 ..\...s. ....
Telnet (telnet), 1 byte
P: 373 D: 373 M: 0

图　7-36

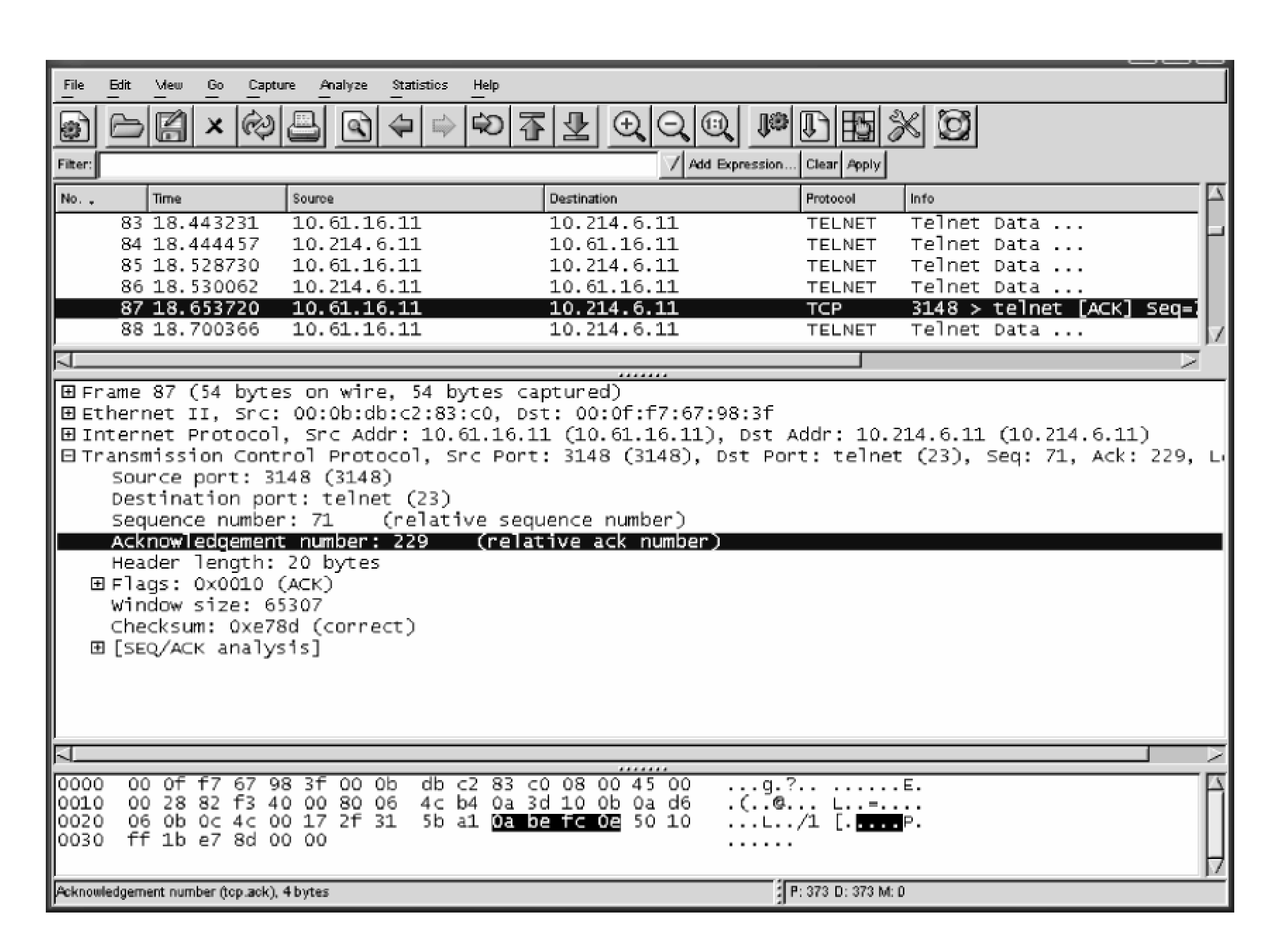

File Edit View Go Capture Analyze Statistics Help
Filter: Add Expression... Clear Apply
No. . Time Source Destination Protocol Info
83 18.443231 10.61.16.11 10.214.6.11 TELNET Telnet Data ...
84 18.444457 10.214.6.11 10.61.16.11 TELNET Telnet Data ...
85 18.528730 10.61.16.11 10.214.6.11 TELNET Telnet Data ...
86 18.530062 10.214.6.11 10.61.16.11 TELNET Telnet Data ...
87 18.653720 10.61.16.11 10.214.6.11 TCP 3148 > telnet [ACK] Seq=
88 18.700366 10.61.16.11 10.214.6.11 TELNET Telnet Data ...
⊞ Frame 87 (54 bytes on wire, 54 bytes captured)
⊞ Ethernet II, Src: 00:0b:db:c2:83:c0, Dst: 00:0f:f7:67:98:3f
⊞ Internet Protocol, Src Addr: 10.61.16.11 (10.61.16.11), Dst Addr: 10.214.6.11 (10.214.6.11)
⊟ Transmission Control Protocol, Src Port: 3148 (3148), Dst Port: telnet (23), Seq: 71, Ack: 229, L
Source port: 3148 (3148)
Destination port: telnet (23)
Sequence number: 71 (relative sequence number)
Acknowledgement number: 229 (relative ack number)
Header length: 20 bytes
⊞ Flags: 0x0010 (ACK)
Window size: 65307
Checksum: 0xe78d (correct)
⊞ [SEQ/ACK analysis]
0000 00 0f f7 67 98 3f 00 0b db c2 83 c0 08 00 45 00 ...g.?.. ......E.
0010 00 28 82 f3 40 00 80 06 4c b4 0a 3d 10 0b 0a d6 .(..@... L..=....
0020 06 0b 0c 4c 00 17 2f 31 5b a1 0a be fc 0e 50 10 ...L../1 [.....P.
0030 ff 1b e7 8d 00 00 ......
Acknowledgement number (tcp.ack), 4 bytes
P: 373 D: 373 M: 0

图　7-37

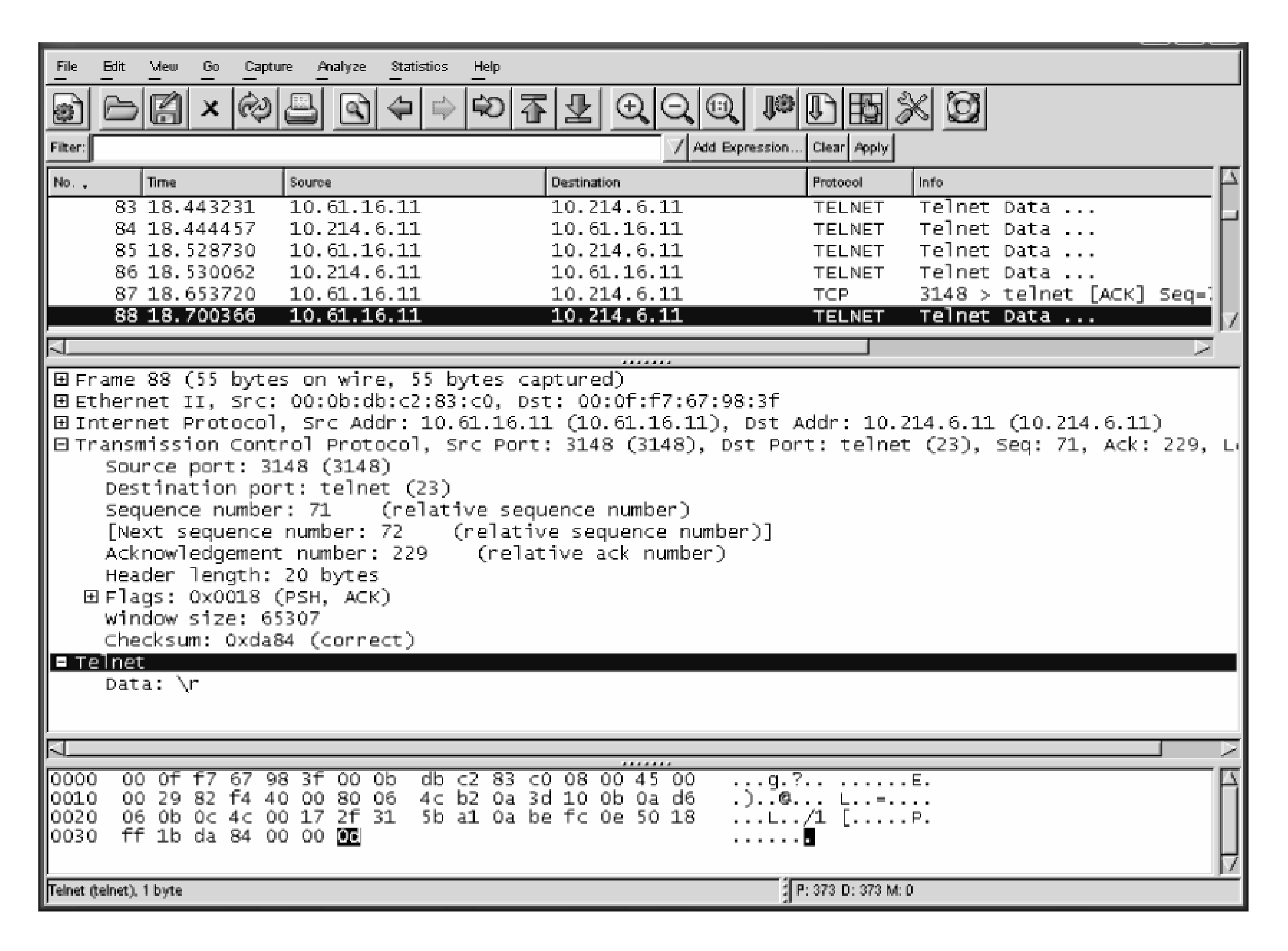

File Edit View Go Capture Analyze Statistics Help
Filter: Add Expression... Clear Apply
No. Time Source Destination Protocol Info
83 18.443231 10.61.16.11 10.214.6.11 TELNET Telnet Data ...
84 18.444457 10.214.6.11 10.61.16.11 TELNET Telnet Data ...
85 18.528730 10.61.16.11 10.214.6.11 TELNET Telnet Data ...
86 18.530062 10.214.6.11 10.61.16.11 TELNET Telnet Data ...
87 18.653720 10.61.16.11 10.214.6.11 TCP 3148 > telnet [ACK] Seq=
88 18.700366 10.61.16.11 10.214.6.11 TELNET Telnet Data ...
Frame 88 (55 bytes on wire, 55 bytes captured)
Ethernet II, Src: 00:0b:db:c2:83:c0, Dst: 00:0f:f7:67:98:3f
Internet Protocol, Src Addr: 10.61.16.11 (10.61.16.11), Dst Addr: 10.214.6.11 (10.214.6.11)
Transmission Control Protocol, Src Port: 3148 (3148), Dst Port: telnet (23), Seq: 71, Ack: 229, L
Source port: 3148 (3148)
Destination port: telnet (23)
Sequence number: 71 (relative sequence number)
[Next sequence number: 72 (relative sequence number)]
Acknowledgement number: 229 (relative ack number)
Header length: 20 bytes
Flags: 0x0018 (PSH, ACK)
Window size: 65307
Checksum: 0xda84 (correct)
Telnet
Data: \r
0000 00 0f f7 67 98 3f 00 0b db c2 83 c0 08 00 45 00 ...g.?.. ......E.
0010 00 29 82 f4 40 00 80 06 4c b2 0a 3d 10 0b 0a d6 .)..@... L..=....
0020 06 0b 0c 4c 00 17 2f 31 5b a1 0a be fc 0e 50 18 ...L../1 [.....P.
0030 ff 1b da 84 00 00 0d .......
Telnet (telnet), 1 byte
P: 373 D: 373 M: 0

图 7-38

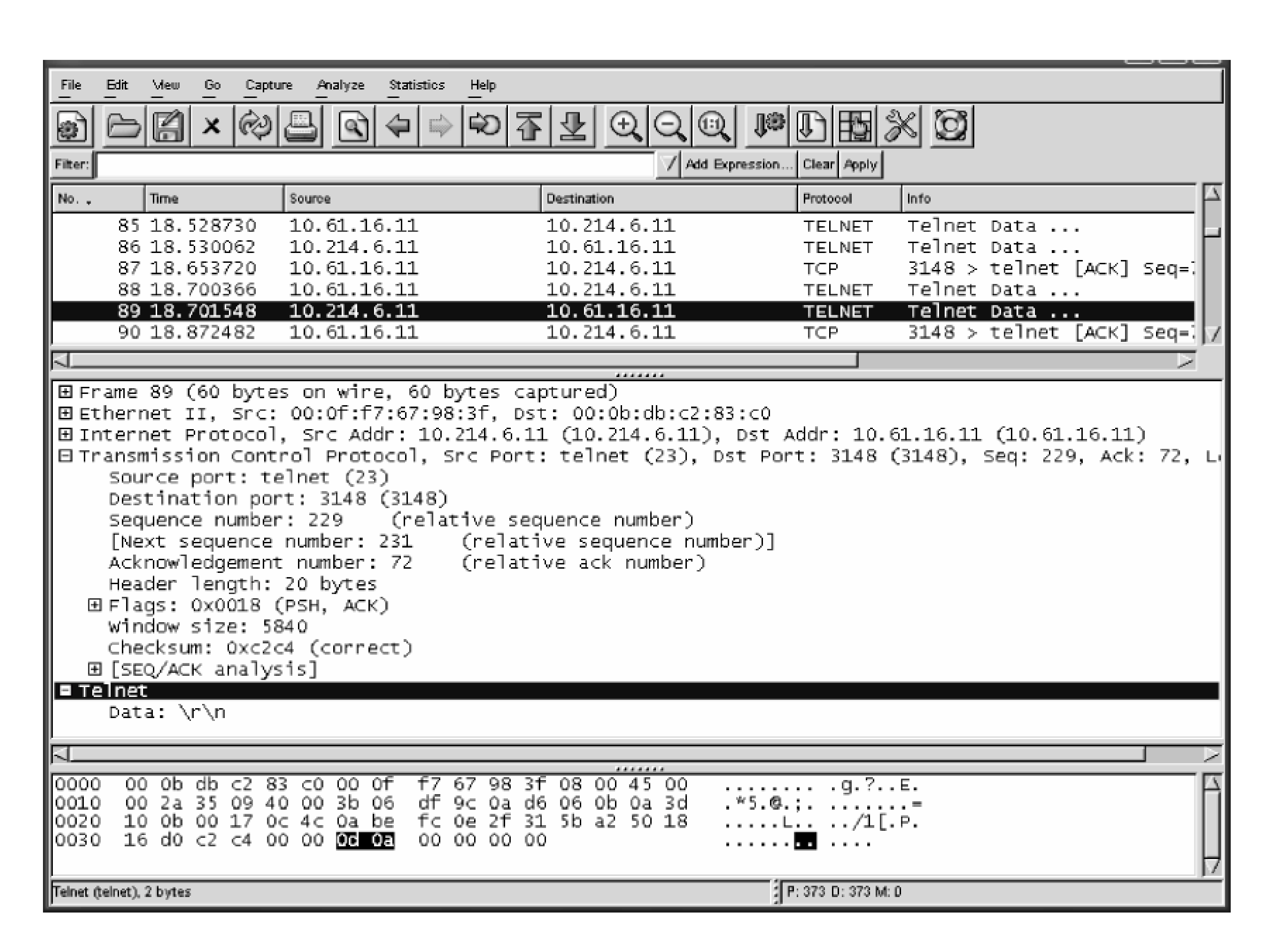

File Edit View Go Capture Analyze Statistics Help
Filter: Add Expression... Clear Apply
No. Time Source Destination Protocol Info
85 18.528730 10.61.16.11 10.214.6.11 TELNET Telnet Data ...
86 18.530062 10.214.6.11 10.61.16.11 TELNET Telnet Data ...
87 18.653720 10.61.16.11 10.214.6.11 TCP 3148 > telnet [ACK] Seq=
88 18.700366 10.61.16.11 10.214.6.11 TELNET Telnet Data ...
89 18.701548 10.214.6.11 10.61.16.11 TELNET Telnet Data ...
90 18.872482 10.61.16.11 10.214.6.11 TCP 3148 > telnet [ACK] Seq=
Frame 89 (60 bytes on wire, 60 bytes captured)
Ethernet II, Src: 00:0f:f7:67:98:3f, Dst: 00:0b:db:c2:83:c0
Internet Protocol, Src Addr: 10.214.6.11 (10.214.6.11), Dst Addr: 10.61.16.11 (10.61.16.11)
Transmission Control Protocol, Src Port: telnet (23), Dst Port: 3148 (3148), Seq: 229, Ack: 72, L
Source port: telnet (23)
Destination port: 3148 (3148)
Sequence number: 229 (relative sequence number)
[Next sequence number: 231 (relative sequence number)]
Acknowledgement number: 72 (relative ack number)
Header length: 20 bytes
Flags: 0x0018 (PSH, ACK)
Window size: 5840
Checksum: 0xc2c4 (correct)
[SEQ/ACK analysis]
Telnet
Data: \r\n
0000 00 0b db c2 83 c0 00 0f f7 67 98 3f 08 00 45 00 ........ .g.?..E.
0010 00 2a 35 09 40 00 3b 06 df 9c 0a d6 06 0b 0a 3d .*5.@.;. .......=
0020 10 0b 00 17 0c 4c 0a be fc 0e 2f 31 5b a2 50 18 .....L.. ../1[.P.
0030 16 d0 c2 c4 00 00 0d 0a 00 00 00 00 ........ ....
Telnet (telnet), 2 bytes
P: 373 D: 373 M: 0

图 7-39

在 Telnet 中"\r\n"是换行命令。

如图 7-40 所示，客户端对收到的报文进行确认。

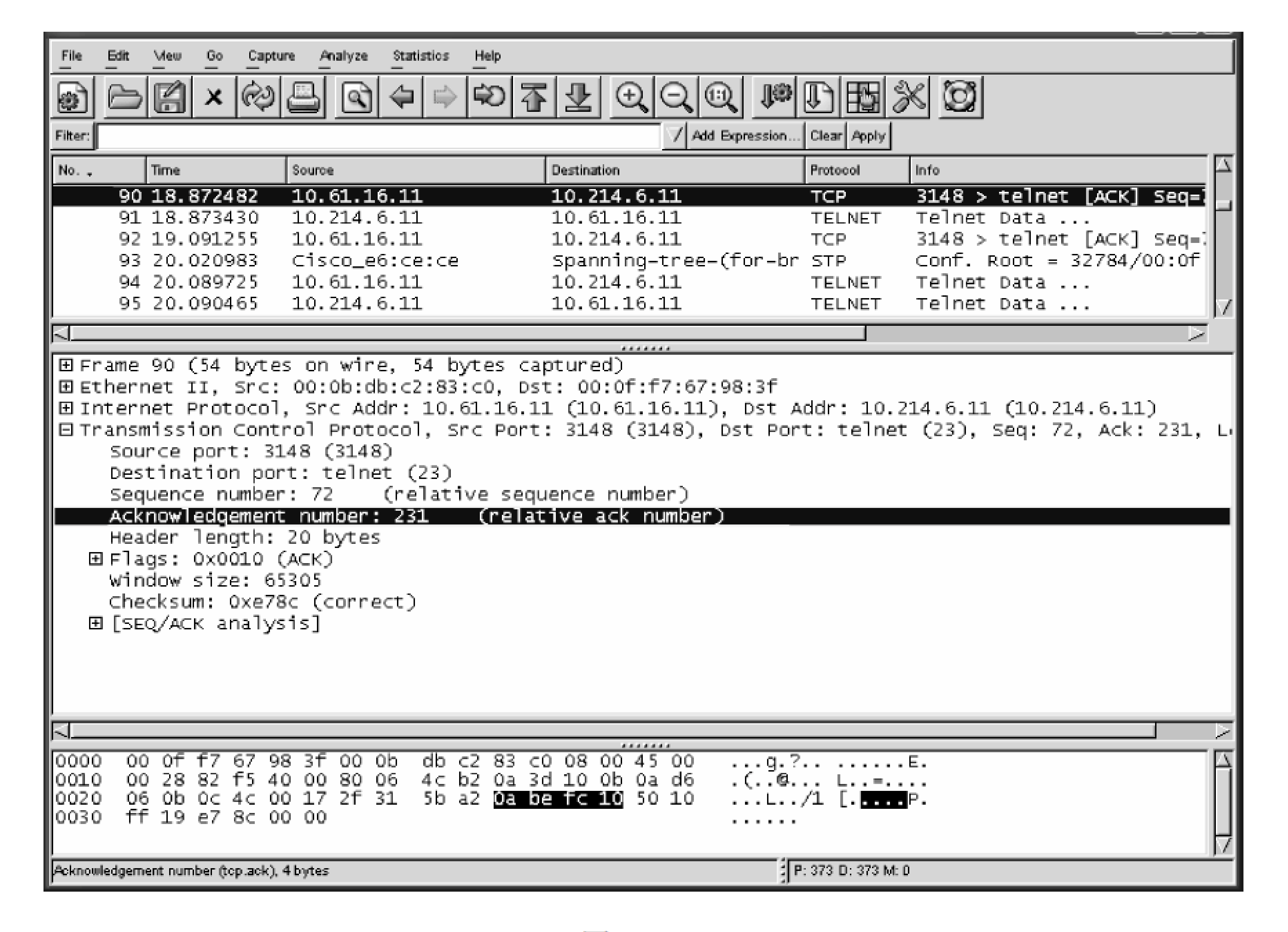



图　7-40

如图 7-41 所示，服务器端发给客户端"ls"命令执行的结果。

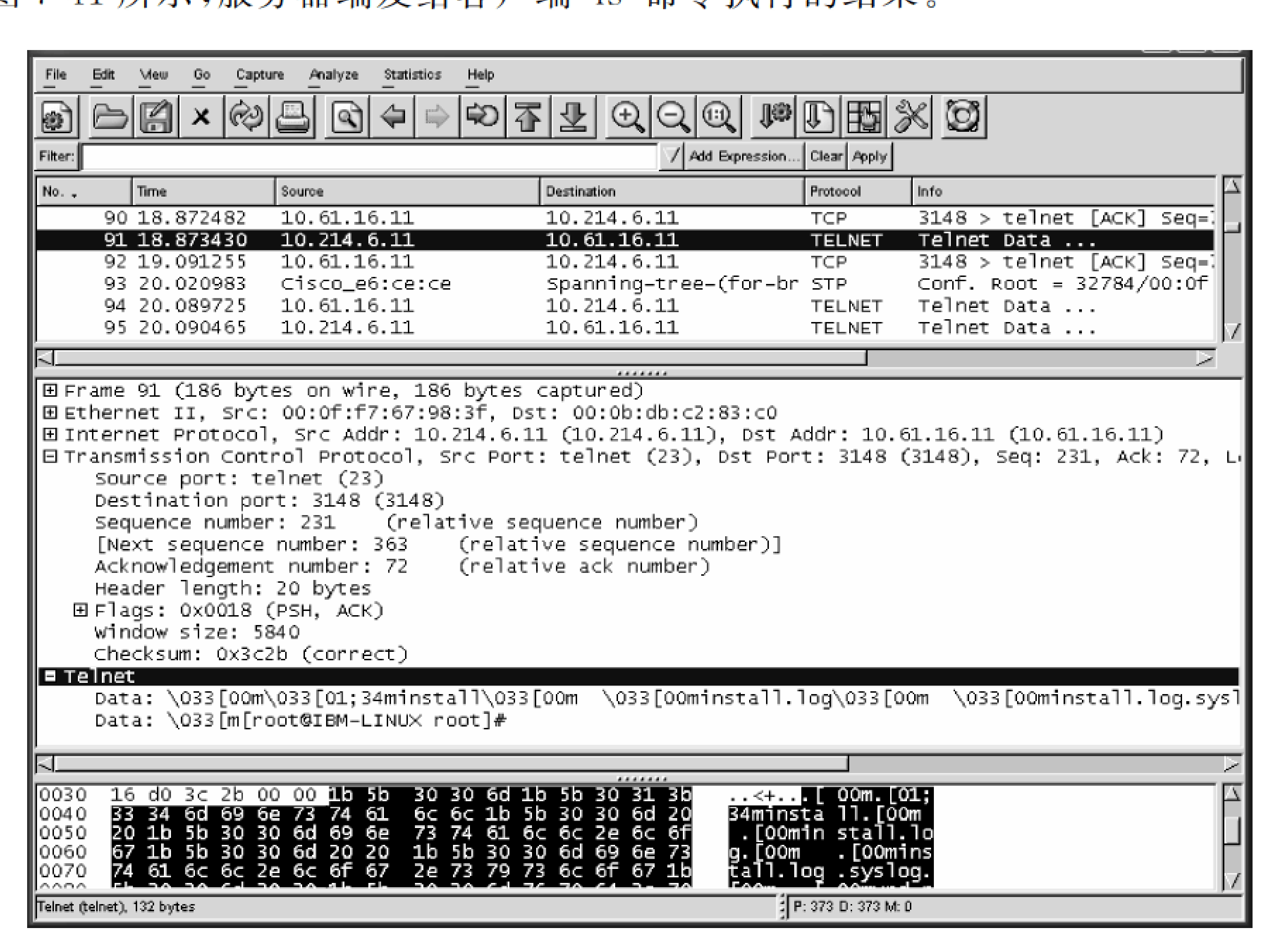



图　7-41

如图 7-42 所示，客户端确认收到的数据。

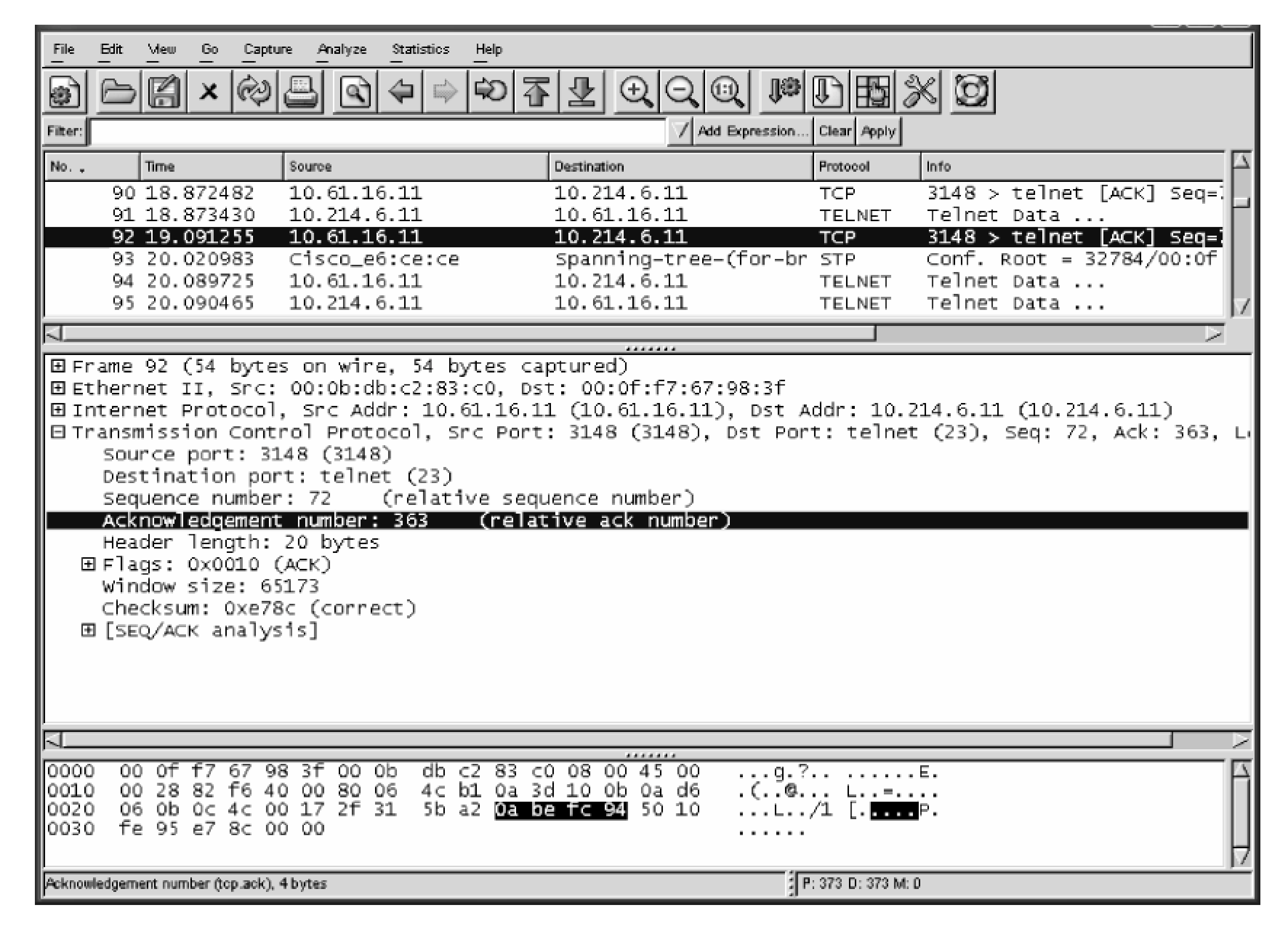



图 7-42

CHAPTER 8

# 第8章 FTP和TFTP协议

## 8.1 FTP协议

FTP(File Transfer Protocol,文件传输协议)是在TCP/IP环境中用来从一台主机向另一台主机传输文件的一个协议,比如从远程FTP服务器上传送文件到本地客户机或从本地客户机传送文件到远程FTP服务器。FTP在OSI协议模型的应用层实现,它在许多计算机系统上采用,是系统间文件传输的公共协议。

FTP采用和HTTP一样的客户机/服务器方式,FTP客户端要在本地安装FTP客户程序。和Telnet一样,FTP需要提供一种登录机制,即需要输入用户名和口令,才能进入远程FTP服务器,与其他客户-服务器模型不同的是,FTP客户与服务器之间用TCP建立了双重连接,一个是控制连接、一个是数据连接。建立双重连接的原因在于FTP是一个交互会话系统,客户端每次调用FTP,便与服务器建立一个会话,会话以控制连接来维持,控制连接负责传输控制信息,例如用户名和口令、改变远程目录的命令、取文件或放回文件的命令,直至退出FTP。当客户端每次请求传送文件时,服务器就与客户建立一个数据连接,进行实际的数据传输。在传送文件数据之前,客户端要通过控制连接来定义文件类型、数据结构和传输方式。一旦数据传输完成,数据连接会话就被关闭,但控制连接依然存在,客户端可以继续发出命令,直到客户端输入CLOSE命令撤销控制连接,再输入QUIT退出FTP会话,此时双方控制进程关闭。

FTP的控制连接使用Telnet协议相同的机制,通过命令和响应来完成,它使用NVT ASCII码,每条命令和响应都是一个短行,每行用回车符和换行符作为结束符。FTP控制命令有User Name(User)、Password(Pass)、Account(Acct)、Change Working Directory(Cwd)、Change To Parent Directory(Cdup)、Structure Mount(Smnt)、Reinitialize(Rein)、Logout(Quit)。

FTP的传输参数命令有Data Port(Port)、Passive(Pasv)、Representation

Type(Type)、File Structure(Stru)、Transfer Mode(Mode)。

和 HTTP 不同的是，FTP 在整个会话期间，服务器必须把控制连接与特定的用户关联起来，随时跟踪用户的状态，跟踪每个活跃的用户会话极大地限制了 FTP 能够同时维护的会话数。无状态的 HTTP 却不必维护任何用户状态信息。

通过下面的 FTP 实例来进一步了解 FTP 协议。

前面讲过 FTP 采用两个 TCP 连接来传输一个文件：

(1) 客户端主动连接服务器，服务器以被动方式打开 21 端口，以建立控制连接。

(2) 文件在客户端与服务器之间传输时，服务器主动打开端口 20，建立一个数据连接。

接下来看看图 8-1，FTP 客户端端口 1364 主动连接服务器 21 端口，在图中可以看到经过 TCP 的三次握手后(3、4、5 行)，建立了控制连接，它们用 1364 和 21 端口来发送控制信息。

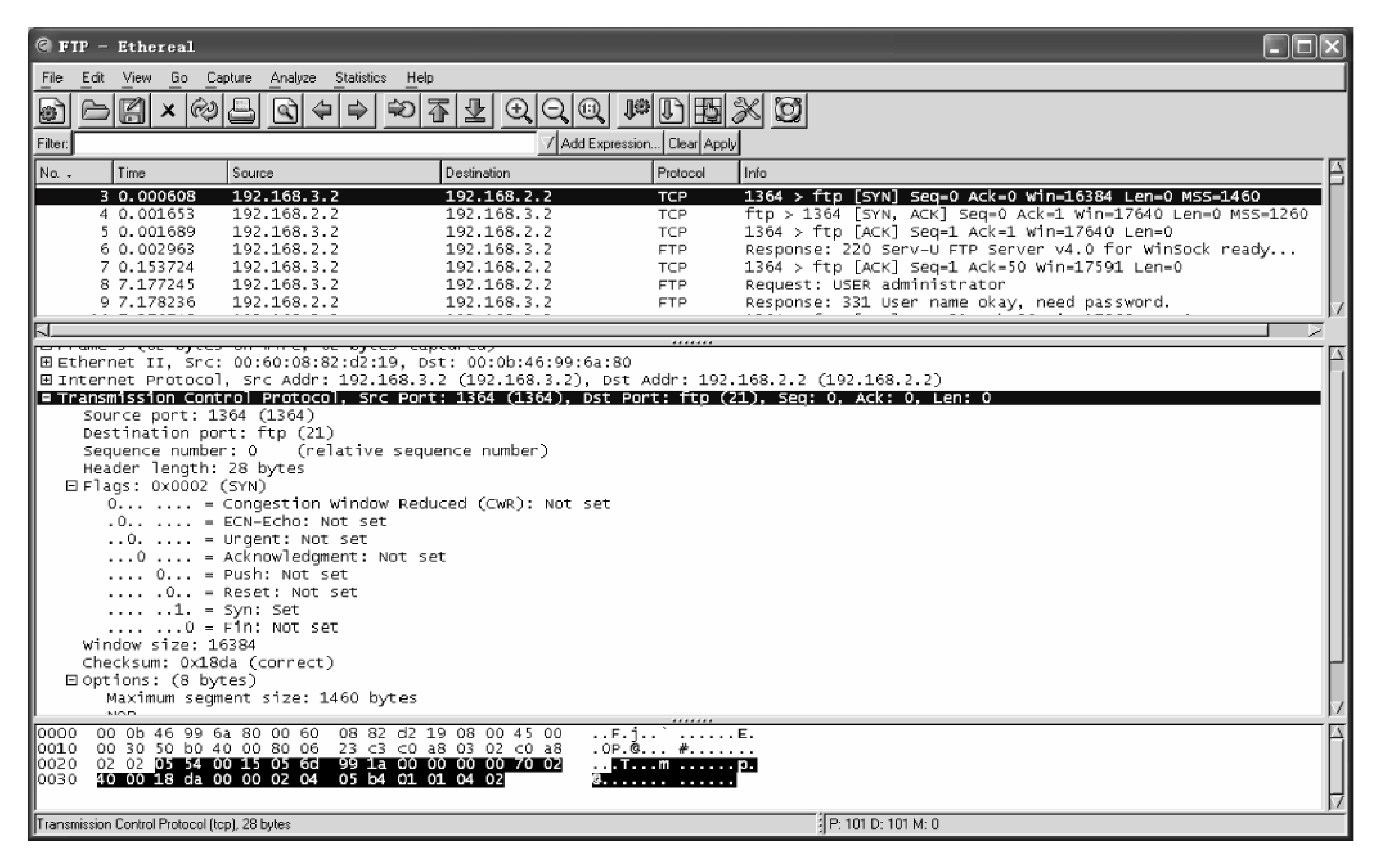



图 8-1

服务器返回客户端 response code 220，如图 8-2 所示，表明服务就绪，服务器准备接受新用户。根据 NVT 标准，要在行结束处使用<CRLF>序列，图中可以看到用\r\n 来表示。

客户端发送一个对报文段 6 的 TCP 确认消息，如图 8-3 所示。

随后，客户端发送一个用户名 USER 命令，后面的 Request arg 为 administrator，也就是用户为 administrator，如图 8-4 所示。

服务器应答，应答号 331 表示用户名被接受，要求输入口令，如图 8-5 所示。

客户端发送一个对报文段 9 的 TCP 确认消息，如图 8-6 所示。

然后，客户端输入密码，如图 8-7 所示，客户端发送 PASS 命令，后面的 Request arg 为 China，也就是登录密码为 China。

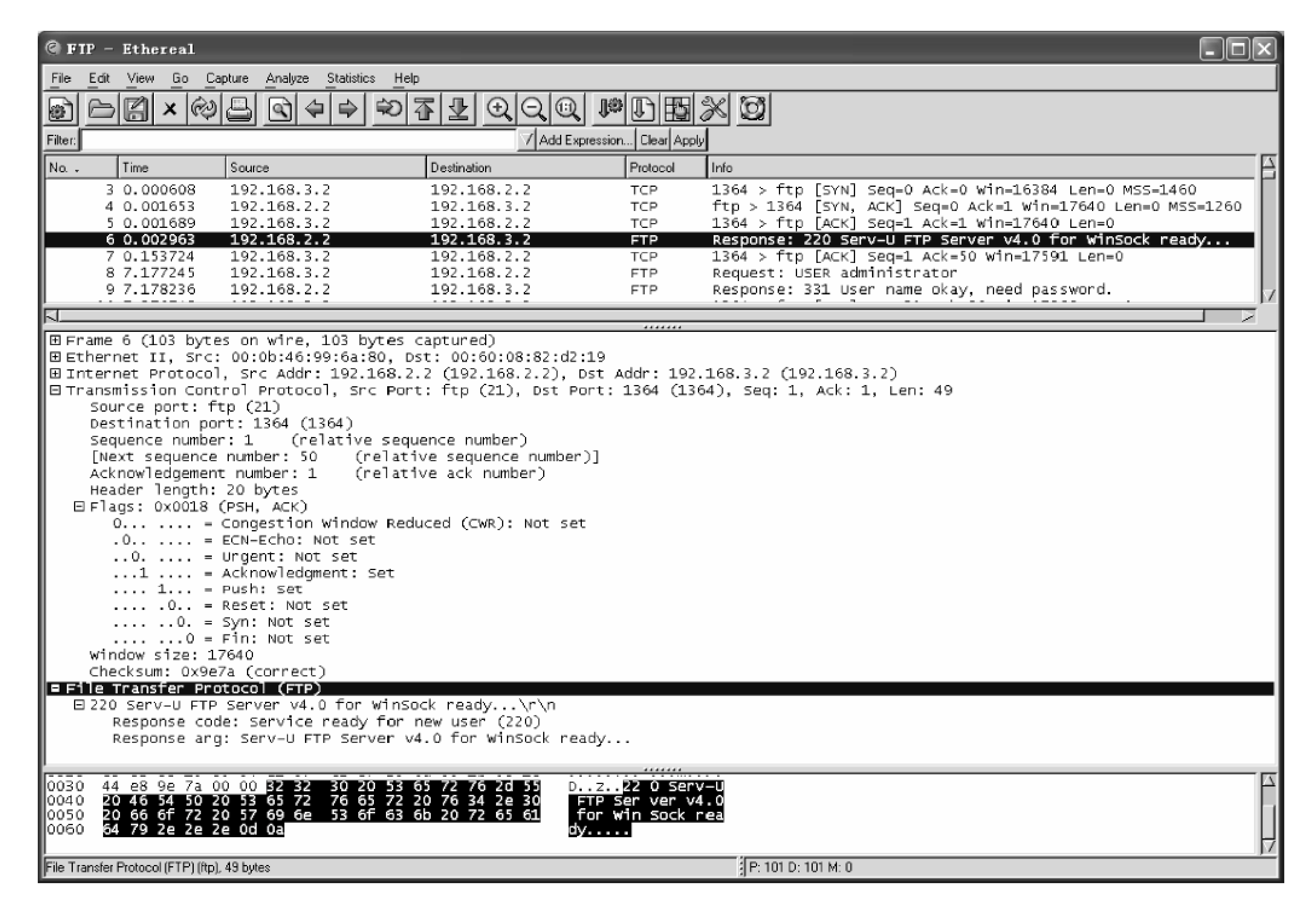

FTP - Ethereal
File Edit View Go Capture Analyze Statistics Help
Filter: Add Expression... Clear Apply
No. Time Source Destination Protocol Info
3 0.000608 192.168.3.2 192.168.2.2 TCP 1364 > ftp [SYN] Seq=0 Ack=0 Win=16384 Len=0 MSS=1460
4 0.001653 192.168.2.2 192.168.3.2 TCP ftp > 1364 [SYN, ACK] Seq=0 Ack=1 Win=17640 Len=0 MSS=1260
5 0.001689 192.168.3.2 192.168.2.2 TCP 1364 > ftp [ACK] Seq=1 Ack=1 Win=17640 Len=0
6 0.002963 192.168.2.2 192.168.3.2 FTP Response: 220 Serv-U FTP Server v4.0 for WinSock ready...
7 0.153724 192.168.3.2 192.168.2.2 TCP 1364 > ftp [ACK] Seq=1 Ack=50 Win=17591 Len=0
8 7.177245 192.168.3.2 192.168.2.2 FTP Request: USER administrator
9 7.178236 192.168.2.2 192.168.3.2 FTP Response: 331 User name okay, need password.
Frame 6 (103 bytes on wire, 103 bytes captured)
Ethernet II, Src: 00:0b:46:99:6a:80, Dst: 00:60:08:82:d2:19
Internet Protocol, Src Addr: 192.168.2.2 (192.168.2.2), Dst Addr: 192.168.3.2 (192.168.3.2)
Transmission Control Protocol, Src Port: ftp (21), Dst Port: 1364 (1364), Seq: 1, Ack: 1, Len: 49
Source port: ftp (21)
Destination port: 1364 (1364)
Sequence number: 1 (relative sequence number)
[Next sequence number: 50 (relative sequence number)]
Acknowledgement number: 1 (relative ack number)
Header length: 20 bytes
Flags: 0x0018 (PSH, ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 1... = Push: Set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 17640
Checksum: 0x9e7a (correct)
File Transfer Protocol (FTP)
220 Serv-U FTP Server v4.0 for WinSock ready...\r\n
Response code: Service ready for new user (220)
Response arg: Serv-U FTP Server v4.0 for WinSock ready...
0030 44 e8 9e 7a 00 00 32 32 30 20 53 65 72 76 2d 55 D..z..22 0 Serv-U
0040 20 46 54 50 20 53 65 72 76 65 72 20 76 34 2e 30 FTP Ser ver v4.0
0050 20 66 6f 72 20 57 69 6e 53 6f 63 6b 20 72 65 61 for Win Sock rea
0060 64 79 2e 2e 2e 0d 0a dy.....
File Transfer Protocol (FTP) (ftp), 49 bytes
P: 101 D: 101 M: 0

图　8-2

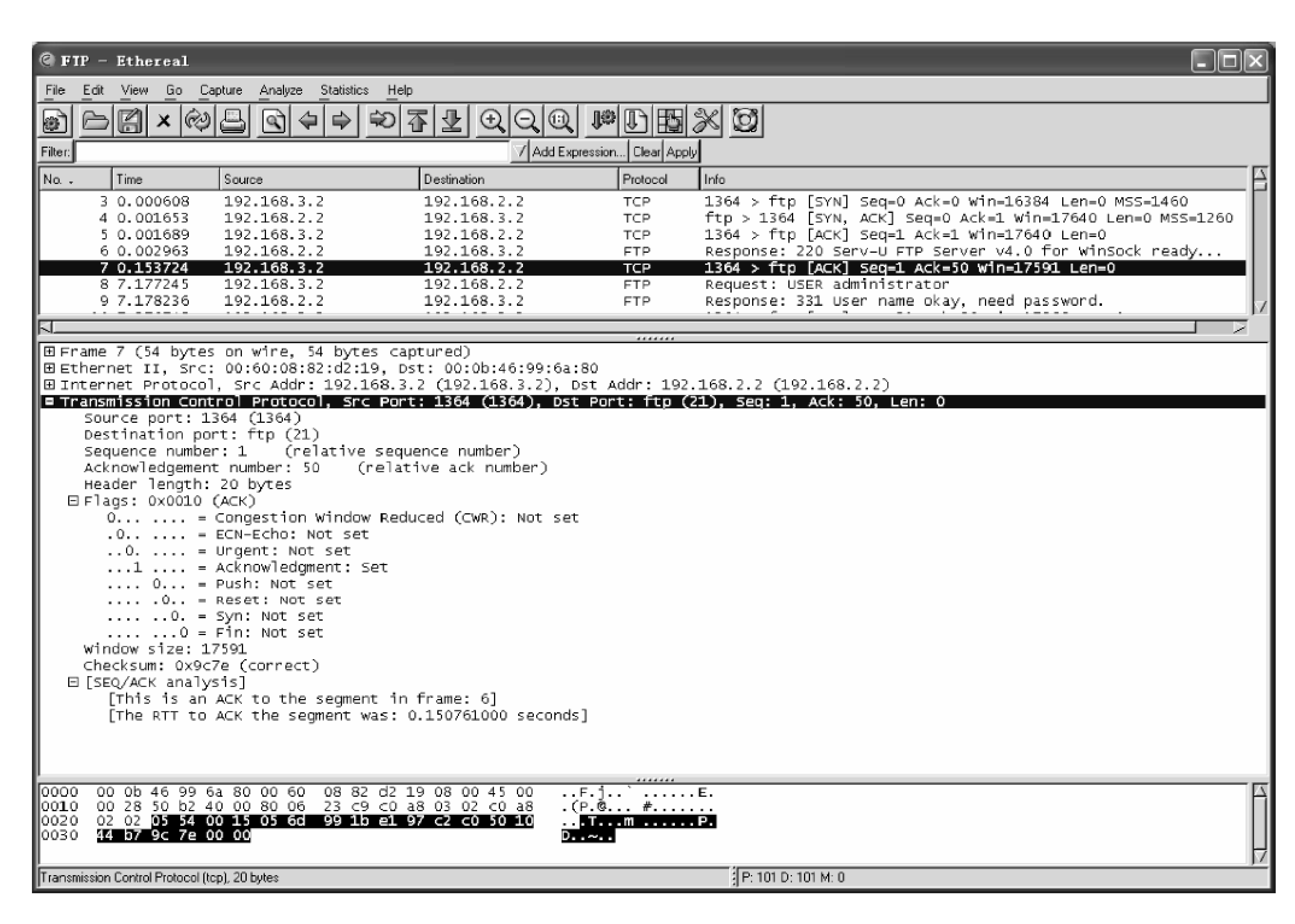

FTP - Ethereal
File Edit View Go Capture Analyze Statistics Help
Filter: Add Expression... Clear Apply
No. Time Source Destination Protocol Info
3 0.000608 192.168.3.2 192.168.2.2 TCP 1364 > ftp [SYN] Seq=0 Ack=0 Win=16384 Len=0 MSS=1460
4 0.001653 192.168.2.2 192.168.3.2 TCP ftp > 1364 [SYN, ACK] Seq=0 Ack=1 Win=17640 Len=0 MSS=1260
5 0.001689 192.168.3.2 192.168.2.2 TCP 1364 > ftp [ACK] Seq=1 Ack=1 Win=17640 Len=0
6 0.002963 192.168.2.2 192.168.3.2 FTP Response: 220 Serv-U FTP Server v4.0 for WinSock ready...
7 0.153724 192.168.3.2 192.168.2.2 TCP 1364 > ftp [ACK] Seq=1 Ack=50 Win=17591 Len=0
8 7.177245 192.168.3.2 192.168.2.2 FTP Request: USER administrator
9 7.178236 192.168.2.2 192.168.3.2 FTP Response: 331 User name okay, need password.
Frame 7 (54 bytes on wire, 54 bytes captured)
Ethernet II, Src: 00:60:08:82:d2:19, Dst: 00:0b:46:99:6a:80
Internet Protocol, Src Addr: 192.168.3.2 (192.168.3.2), Dst Addr: 192.168.2.2 (192.168.2.2)
Transmission Control Protocol, Src Port: 1364 (1364), Dst Port: ftp (21), Seq: 1, Ack: 50, Len: 0
Source port: 1364 (1364)
Destination port: ftp (21)
Sequence number: 1 (relative sequence number)
Acknowledgement number: 50 (relative ack number)
Header length: 20 bytes
Flags: 0x0010 (ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 0... = Push: Not set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 17591
Checksum: 0x9c7e (correct)
[SEQ/ACK analysis]
[This is an ACK to the segment in frame: 6]
[The RTT to ACK the segment was: 0.150761000 seconds]
0000 00 0b 46 99 6a 80 00 60 08 82 d2 19 08 00 45 00 ..F.j..` ......E.
0010 00 28 50 b2 40 00 80 06 23 c9 c0 a8 03 02 c0 a8 .(P.@... #.......
0020 02 02 05 54 00 15 05 6d 99 1b e1 97 c2 c0 50 10 ...T...m ......P.
0030 44 b7 9c 7e 00 00 D..~..
Transmission Control Protocol (tcp), 20 bytes
P: 101 D: 101 M: 0

图　8-3

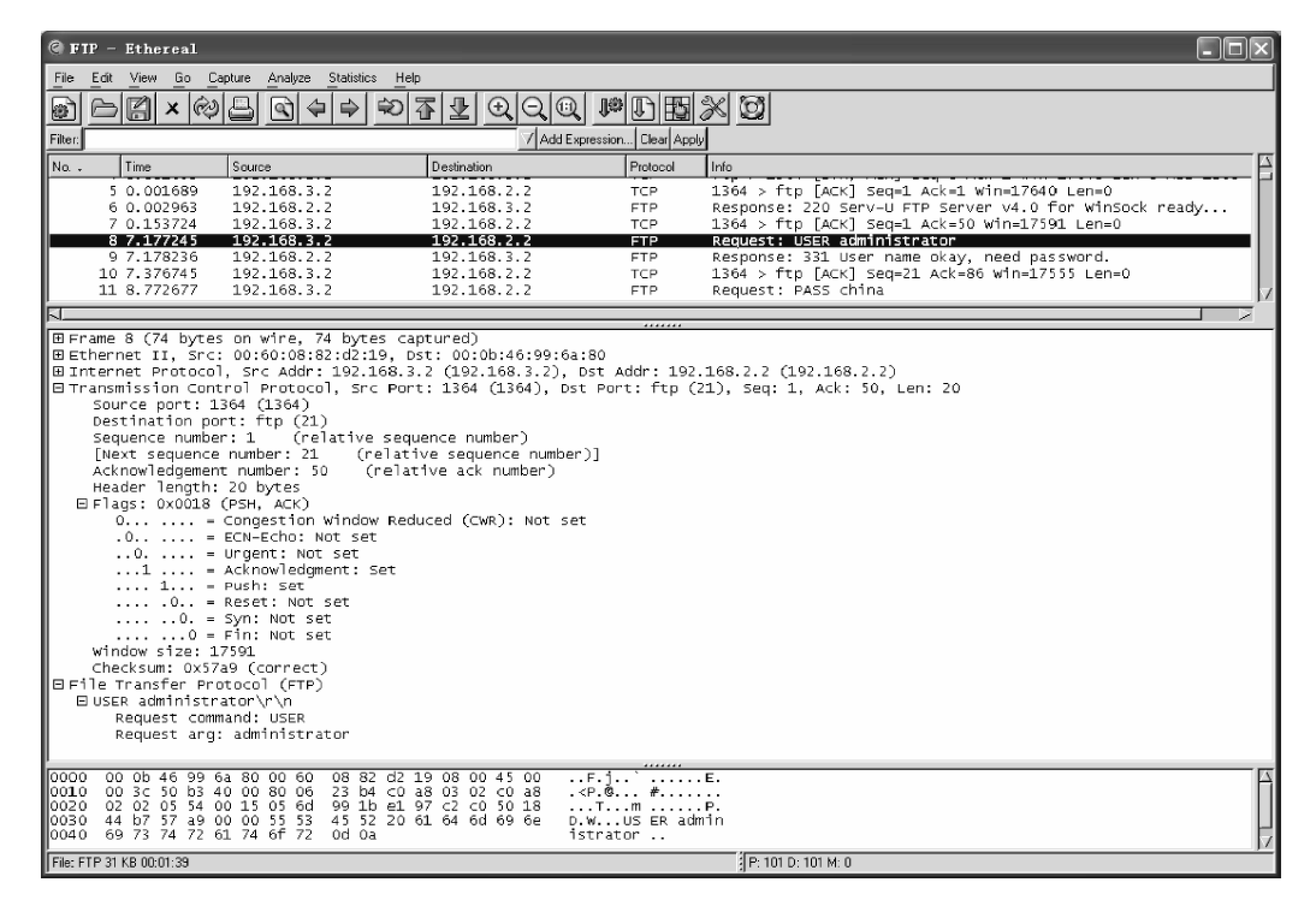

FTP - Ethereal
File Edit View Go Capture Analyze Statistics Help
Filter: Add Expression... Clear Apply
No. Time Source Destination Protocol Info
5 0.001689 192.168.3.2 192.168.2.2 TCP 1364 > ftp [ACK] Seq=1 Ack=1 Win=17640 Len=0
6 0.002963 192.168.2.2 192.168.3.2 FTP Response: 220 Serv-U FTP Server v4.0 for WinSock ready...
7 0.153724 192.168.3.2 192.168.2.2 TCP 1364 > ftp [ACK] Seq=1 Ack=50 Win=17591 Len=0
8 7.177245 192.168.3.2 192.168.2.2 FTP Request: USER administrator
9 7.178236 192.168.2.2 192.168.3.2 FTP Response: 331 User name okay, need password.
10 7.376745 192.168.3.2 192.168.2.2 TCP 1364 > ftp [ACK] Seq=21 Ack=86 Win=17555 Len=0
11 8.772677 192.168.3.2 192.168.2.2 FTP Request: PASS china
Frame 8 (74 bytes on wire, 74 bytes captured)
Ethernet II, Src: 00:60:08:82:d2:19, Dst: 00:0b:46:99:6a:80
Internet Protocol, Src Addr: 192.168.3.2 (192.168.3.2), Dst Addr: 192.168.2.2 (192.168.2.2)
Transmission Control Protocol, Src Port: 1364 (1364), Dst Port: ftp (21), Seq: 1, Ack: 50, Len: 20
Source port: 1364 (1364)
Destination port: ftp (21)
Sequence number: 1 (relative sequence number)
[Next sequence number: 21 (relative sequence number)]
Acknowledgement number: 50 (relative ack number)
Header length: 20 bytes
Flags: 0x0018 (PSH, ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 1... = Push: Set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 17591
Checksum: 0x57a9 (correct)
File Transfer Protocol (FTP)
USER administrator\r\n
Request command: USER
Request arg: administrator
0000 00 0b 46 99 6a 80 00 60 08 82 d2 19 08 00 45 00
0010 00 3c 50 b3 40 00 80 06 23 b4 c0 a8 03 02 c0 a8
0020 02 02 05 54 00 15 05 6d 99 1b e1 97 c2 c0 50 18
0030 44 b7 57 a9 00 00 55 53 45 52 20 61 64 6d 69 6e
0040 69 73 74 72 61 74 6f 72 0d 0a
File: FTP 31 KB 00:01:39
P: 101 D: 101 M: 0

图 8-4

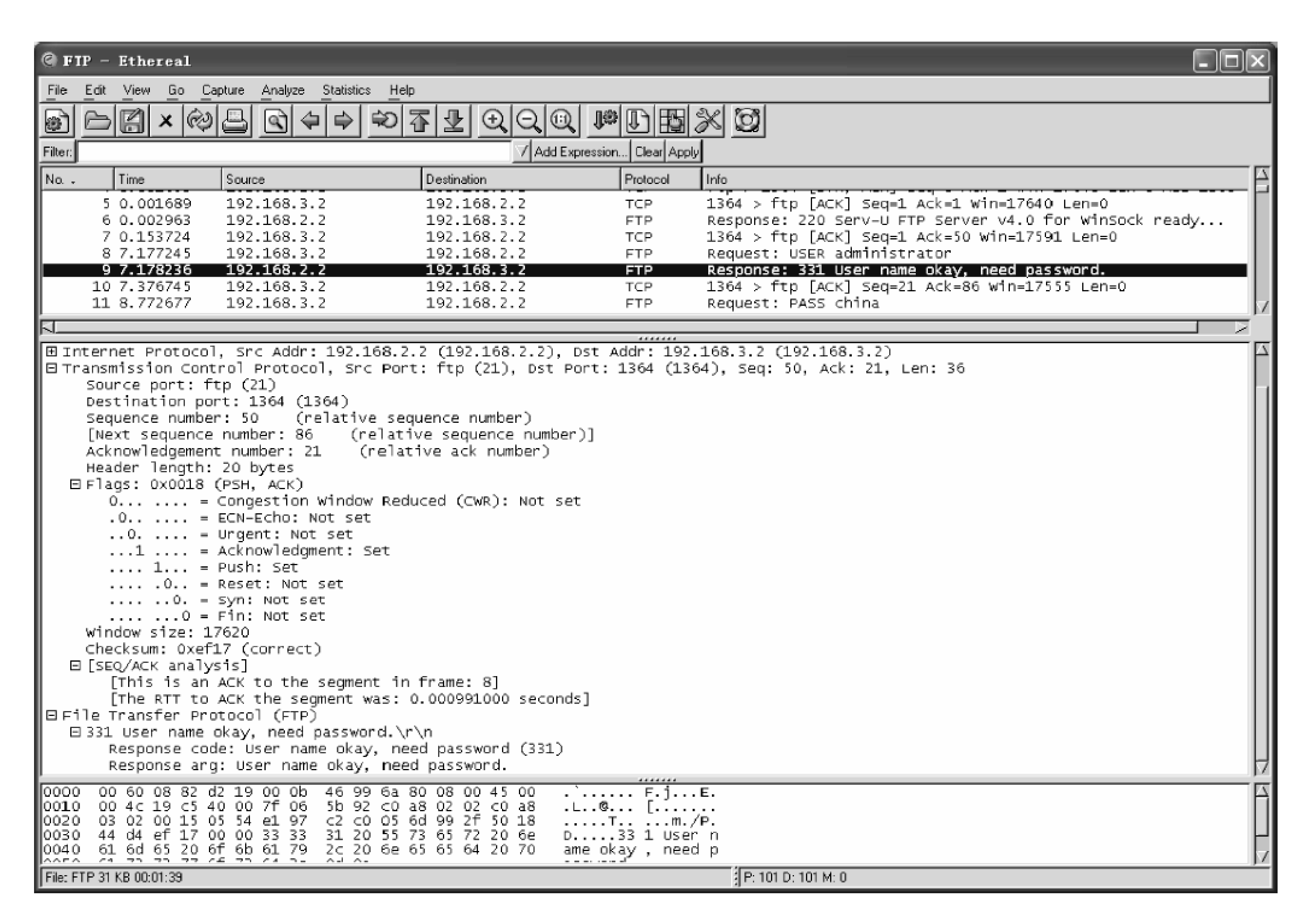

FTP - Ethereal
File Edit View Go Capture Analyze Statistics Help
Filter: Add Expression... Clear Apply
No. Time Source Destination Protocol Info
5 0.001689 192.168.3.2 192.168.2.2 TCP 1364 > ftp [ACK] Seq=1 Ack=1 Win=17640 Len=0
6 0.002963 192.168.2.2 192.168.3.2 FTP Response: 220 Serv-U FTP Server v4.0 for WinSock ready...
7 0.153724 192.168.3.2 192.168.2.2 TCP 1364 > ftp [ACK] Seq=1 Ack=50 Win=17591 Len=0
8 7.177245 192.168.3.2 192.168.2.2 FTP Request: USER administrator
9 7.178236 192.168.2.2 192.168.3.2 FTP Response: 331 User name okay, need password.
10 7.376745 192.168.3.2 192.168.2.2 TCP 1364 > ftp [ACK] Seq=21 Ack=86 Win=17555 Len=0
11 8.772677 192.168.3.2 192.168.2.2 FTP Request: PASS china
Internet Protocol, Src Addr: 192.168.2.2 (192.168.2.2), Dst Addr: 192.168.3.2 (192.168.3.2)
Transmission Control Protocol, Src Port: ftp (21), Dst Port: 1364 (1364), Seq: 50, Ack: 21, Len: 36
Source port: ftp (21)
Destination port: 1364 (1364)
Sequence number: 50 (relative sequence number)
[Next sequence number: 86 (relative sequence number)]
Acknowledgement number: 21 (relative ack number)
Header length: 20 bytes
Flags: 0x0018 (PSH, ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 1... = Push: Set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 17620
Checksum: 0xef17 (correct)
[SEQ/ACK analysis]
[This is an ACK to the segment in frame: 8]
[The RTT to ACK the segment was: 0.000991000 seconds]
File Transfer Protocol (FTP)
331 User name okay, need password.\r\n
Response code: User name okay, need password (331)
Response arg: User name okay, need password.
0000 00 60 08 82 d2 19 00 0b 46 99 6a 80 08 00 45 00
0010 00 4c 19 c5 40 00 7f 06 5b 92 c0 a8 02 02 c0 a8
0020 03 02 00 15 05 54 e1 97 c2 c0 05 6d 99 2f 50 18
0030 44 d4 ef 17 00 00 33 33 31 20 55 73 65 72 20 6e
0040 61 6d 65 20 6f 6b 61 79 2c 20 6e 65 65 64 20 70
File: FTP 31 KB 00:01:39
P: 101 D: 101 M: 0

图 8-5

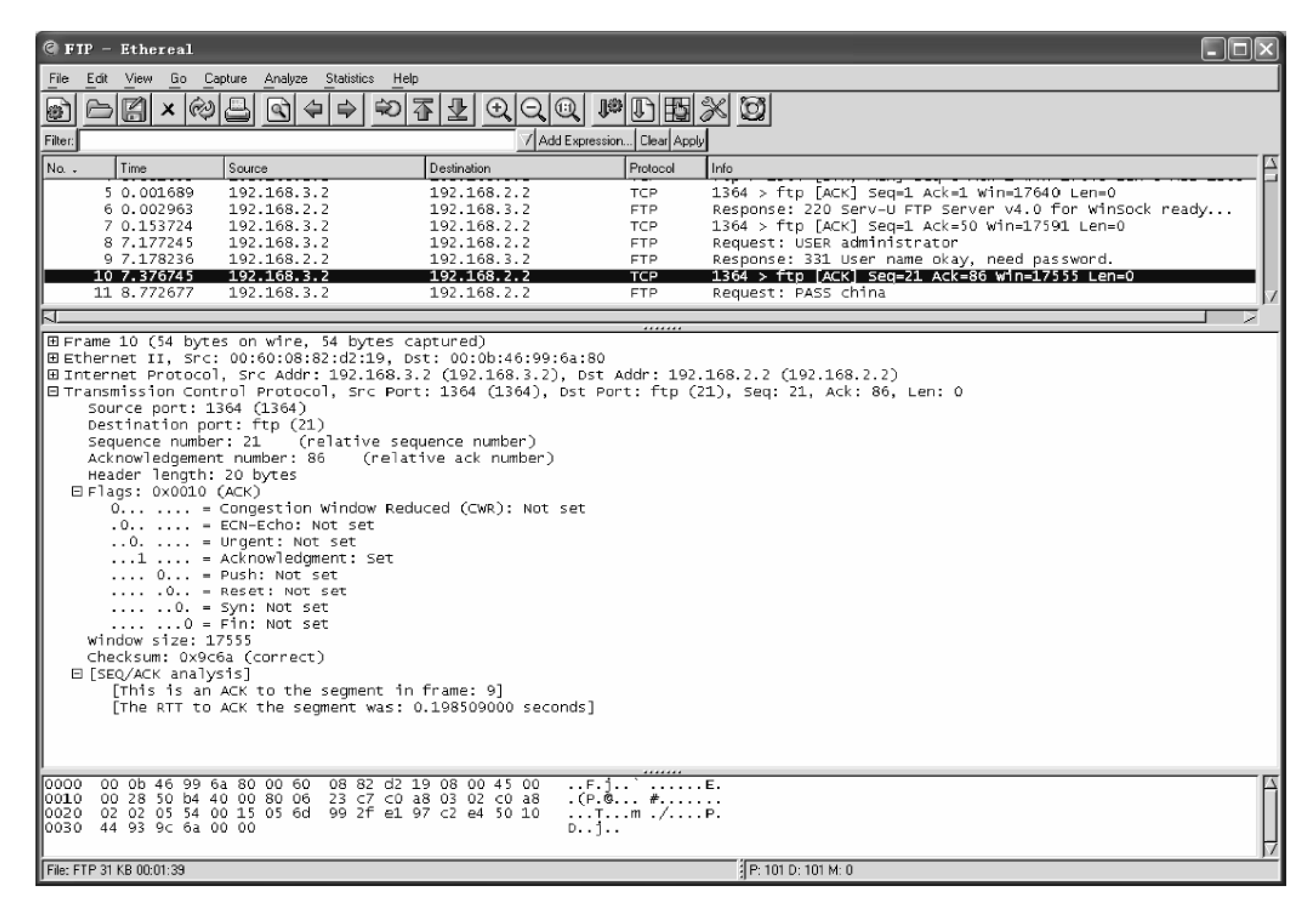

FTP - Ethereal
File Edit View Go Capture Analyze Statistics Help
Filter: Add Expression... Clear Apply
No. Time Source Destination Protocol Info
5 0.001689 192.168.3.2 192.168.2.2 TCP 1364 > ftp [ACK] Seq=1 Ack=1 Win=17640 Len=0
6 0.002963 192.168.2.2 192.168.3.2 FTP Response: 220 Serv-U FTP Server v4.0 for WinSock ready...
7 0.153724 192.168.3.2 192.168.2.2 TCP 1364 > ftp [ACK] Seq=1 Ack=50 Win=17591 Len=0
8 7.177245 192.168.3.2 192.168.2.2 FTP Request: USER administrator
9 7.178236 192.168.2.2 192.168.3.2 FTP Response: 331 User name okay, need password.
10 7.376745 192.168.3.2 192.168.2.2 TCP 1364 > ftp [ACK] Seq=21 Ack=86 Win=17555 Len=0
11 8.772677 192.168.3.2 192.168.2.2 FTP Request: PASS china
Frame 10 (54 bytes on wire, 54 bytes captured)
Ethernet II, Src: 00:60:08:82:d2:19, Dst: 00:0b:46:99:6a:80
Internet Protocol, Src Addr: 192.168.3.2 (192.168.3.2), Dst Addr: 192.168.2.2 (192.168.2.2)
Transmission Control Protocol, Src Port: 1364 (1364), Dst Port: ftp (21), Seq: 21, Ack: 86, Len: 0
Source port: 1364 (1364)
Destination port: ftp (21)
Sequence number: 21 (relative sequence number)
Acknowledgement number: 86 (relative ack number)
Header length: 20 bytes
Flags: 0x0010 (ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 0... = Push: Not set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 17555
Checksum: 0x9c6a (correct)
[SEQ/ACK analysis]
[This is an ACK to the segment in frame: 9]
[The RTT to ACK the segment was: 0.198509000 seconds]
0000 00 0b 46 99 6a 80 00 60 08 82 d2 19 08 00 45 00 ..F.j..` ......E.
0010 00 28 50 b4 40 00 80 06 23 c7 c0 a8 03 02 c0 a8 .(P.@... #.......
0020 02 02 05 54 00 15 05 6d 99 2f e1 97 c2 e4 50 10 ...T...m ./....P.
0030 44 93 9c 6a 00 00 D..j..
File: FTP 31 KB 00:01:39
P: 101 D: 101 M: 0

图　8-6

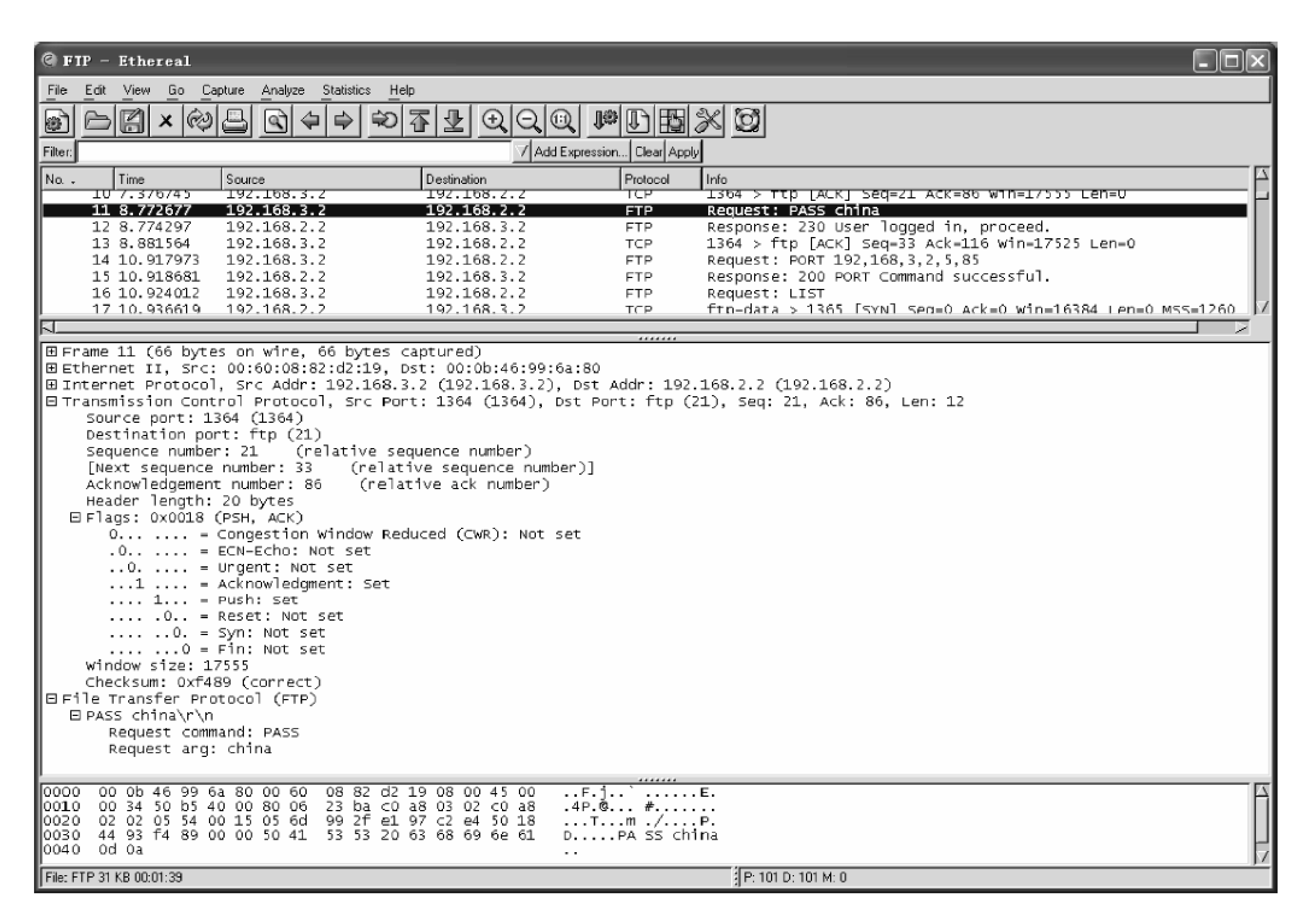

FTP - Ethereal
File Edit View Go Capture Analyze Statistics Help
Filter: Add Expression... Clear Apply
No. Time Source Destination Protocol Info
11 8.772677 192.168.3.2 192.168.2.2 FTP Request: PASS china
12 8.774297 192.168.2.2 192.168.3.2 FTP Response: 230 User logged in, proceed.
13 8.881564 192.168.3.2 192.168.2.2 TCP 1364 > ftp [ACK] Seq=33 Ack=116 Win=17525 Len=0
14 10.917973 192.168.3.2 192.168.2.2 FTP Request: PORT 192,168,3,2,5,85
15 10.918681 192.168.2.2 192.168.3.2 FTP Response: 200 PORT Command successful.
16 10.924012 192.168.3.2 192.168.2.2 FTP Request: LIST
Frame 11 (66 bytes on wire, 66 bytes captured)
Ethernet II, Src: 00:60:08:82:d2:19, Dst: 00:0b:46:99:6a:80
Internet Protocol, Src Addr: 192.168.3.2 (192.168.3.2), Dst Addr: 192.168.2.2 (192.168.2.2)
Transmission Control Protocol, Src Port: 1364 (1364), Dst Port: ftp (21), Seq: 21, Ack: 86, Len: 12
Source port: 1364 (1364)
Destination port: ftp (21)
Sequence number: 21 (relative sequence number)
[Next sequence number: 33 (relative sequence number)]
Acknowledgement number: 86 (relative ack number)
Header length: 20 bytes
Flags: 0x0018 (PSH, ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 1... = Push: Set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 17555
Checksum: 0xf489 (correct)
File Transfer Protocol (FTP)
PASS china\r\n
Request command: PASS
Request arg: china
0000 00 0b 46 99 6a 80 00 60 08 82 d2 19 08 00 45 00 ..F.j..` ......E.
0010 00 34 50 b5 40 00 80 06 23 ba c0 a8 03 02 c0 a8 .4P.@... #.......
0020 02 02 05 54 00 15 05 6d 99 2f e1 97 c2 e4 50 18 ...T...m ./....P.
0030 44 93 f4 89 00 00 50 41 53 53 20 63 68 69 6e 61 D.....PA SS china
0040 0d 0a ..
File: FTP 31 KB 00:01:39
P: 101 D: 101 M: 0

图　8-7

服务器端产生一个应答，应答号 230 表示用户成功登录，如图 8-8 所示。

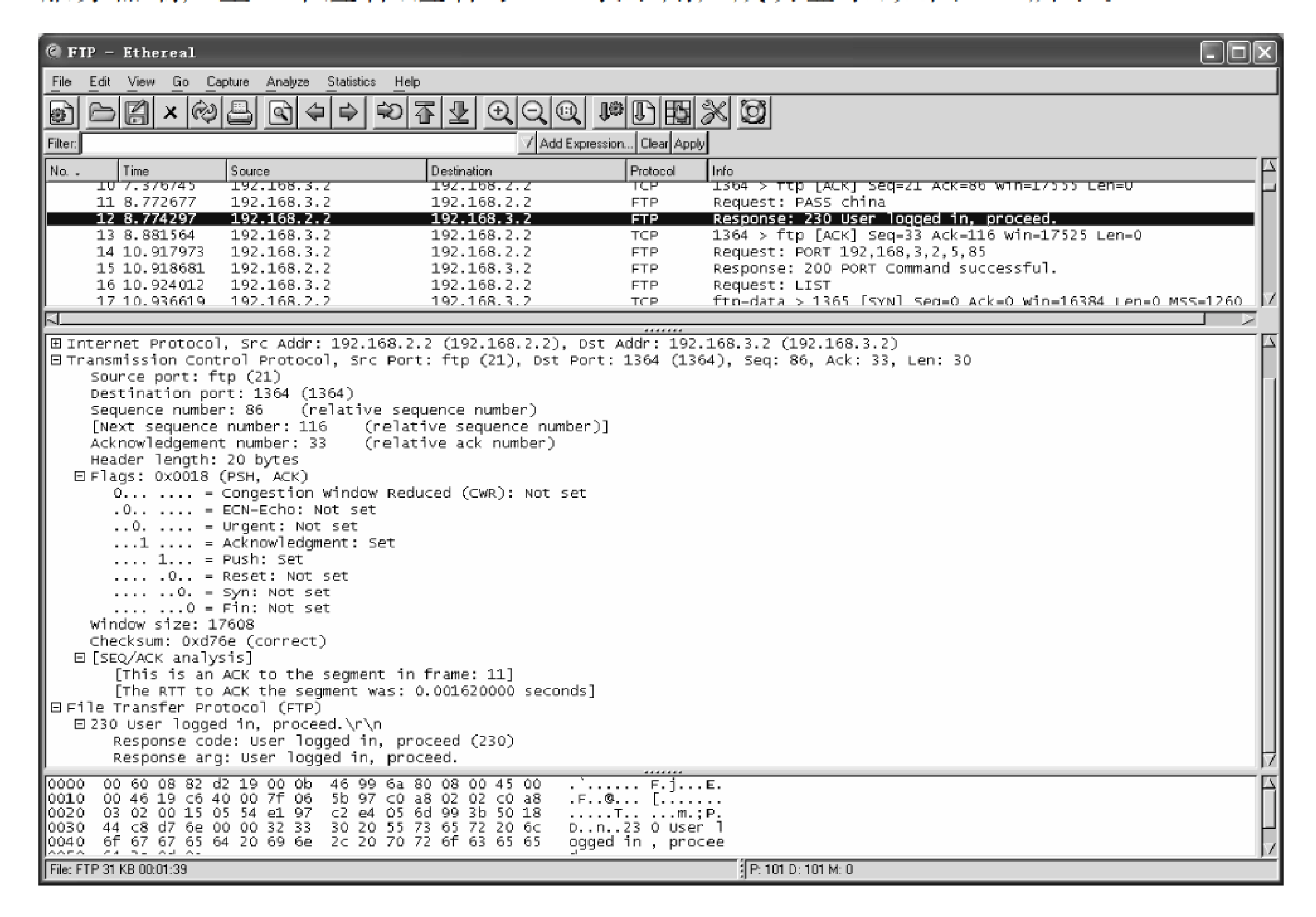



图 8-8

客户端发送一个对报文段 12 的 TCP 确认消息，如图 8-9 所示。

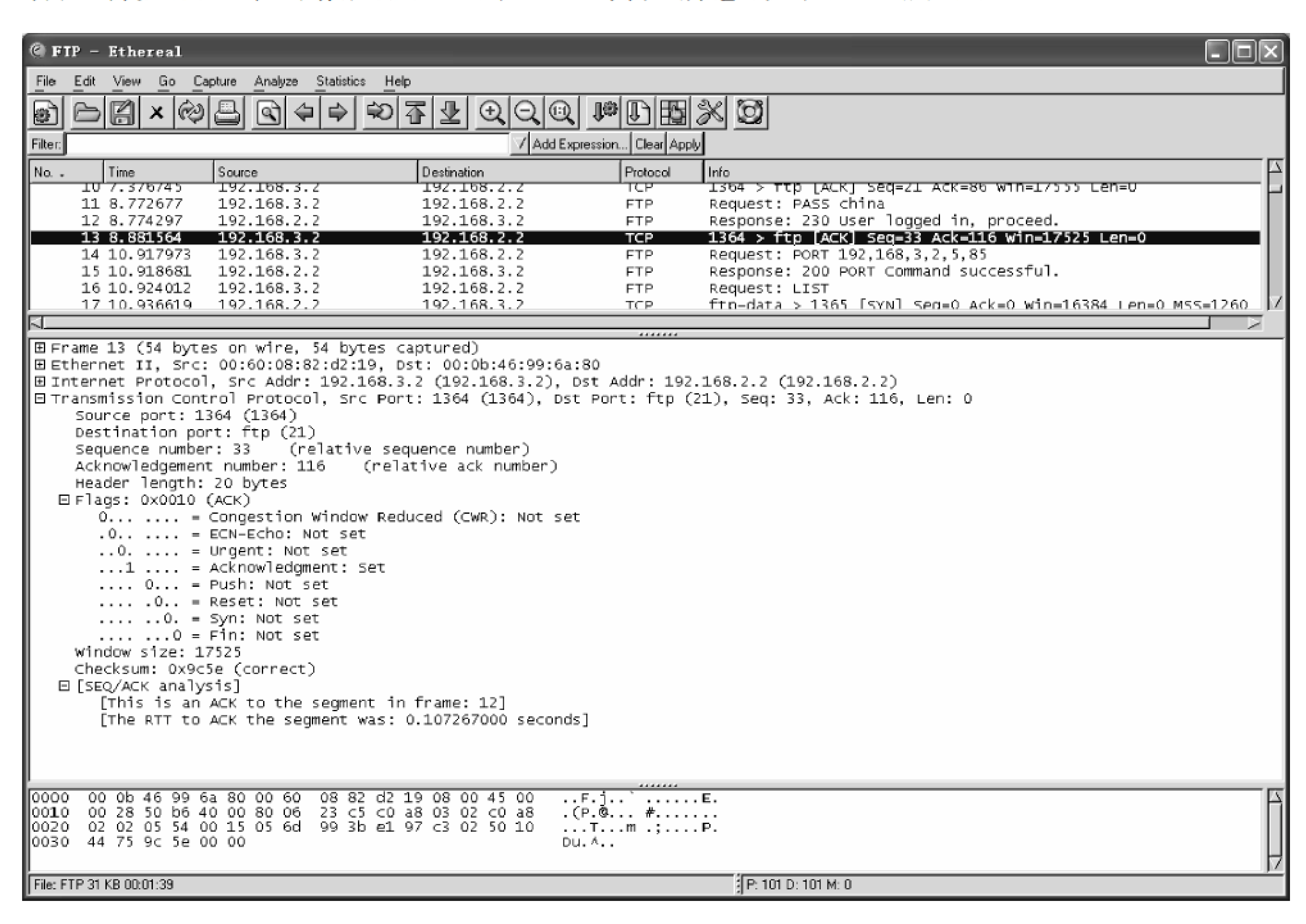



图 8-9

客户端发送一个 PORT 命令，如图 8-10 所示，可以看到这个命令为 PORT 192,168,3,2,5,85。这个命令包含了两个部分，一个部分客户端的 IP 地址即 192.168.3.2，第二个部分即为服务器用 20 端口主动打开数据连接时客户端使用的端口，(5,85)这两个数字转换成端口：5 * 256+85=1365。

**注意**：这种客户端发送 PORT 命令的方式称主动方式。FTP 还有一种方式叫被动方式。

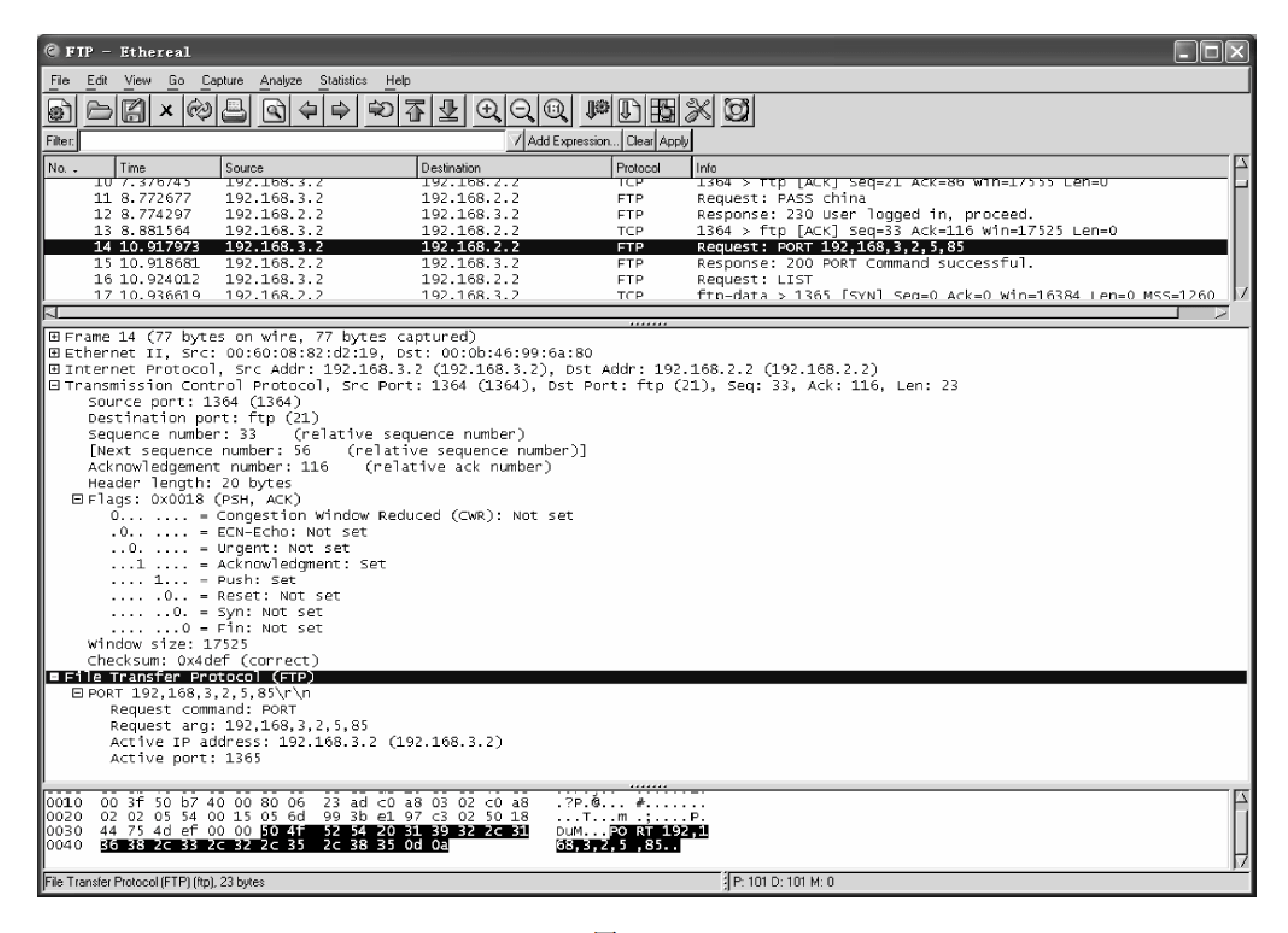



图　8-10

服务器端返回一个应答，应答号 200 表示 PORT 命令被接受，如图 8-11 所示。

客户端发送一个 LIST 请求命令，请求一个列表显示文件或目录，如图 8-12 所示。

服务器端用端口 20 发送到客户端端口 1365 一个同步信息。注意，这里就开始建立数据连接了，可以看到服务器的端口为 20，如图 8-13 所示。

客户端发送一个同步确认信息，如图 8-14 所示。

服务器端对报文段 16 即 LIST 命令的回应，如图 8-15 所示，可看到数据传输使用 ASCII 码模式。

服务端发送一个确认信息，到此经过 TCP 3 次握手数据连接已经建立，如图 8-16 所示。

以下三个图片为服务器端用端口 20 对 LIST 命令的执行结果进行数据传输，如图 8-17～图 8-19 所示。

客户端对收到的报文段(21、22、23)进行 TCP 确认，如图 8-20 所示。

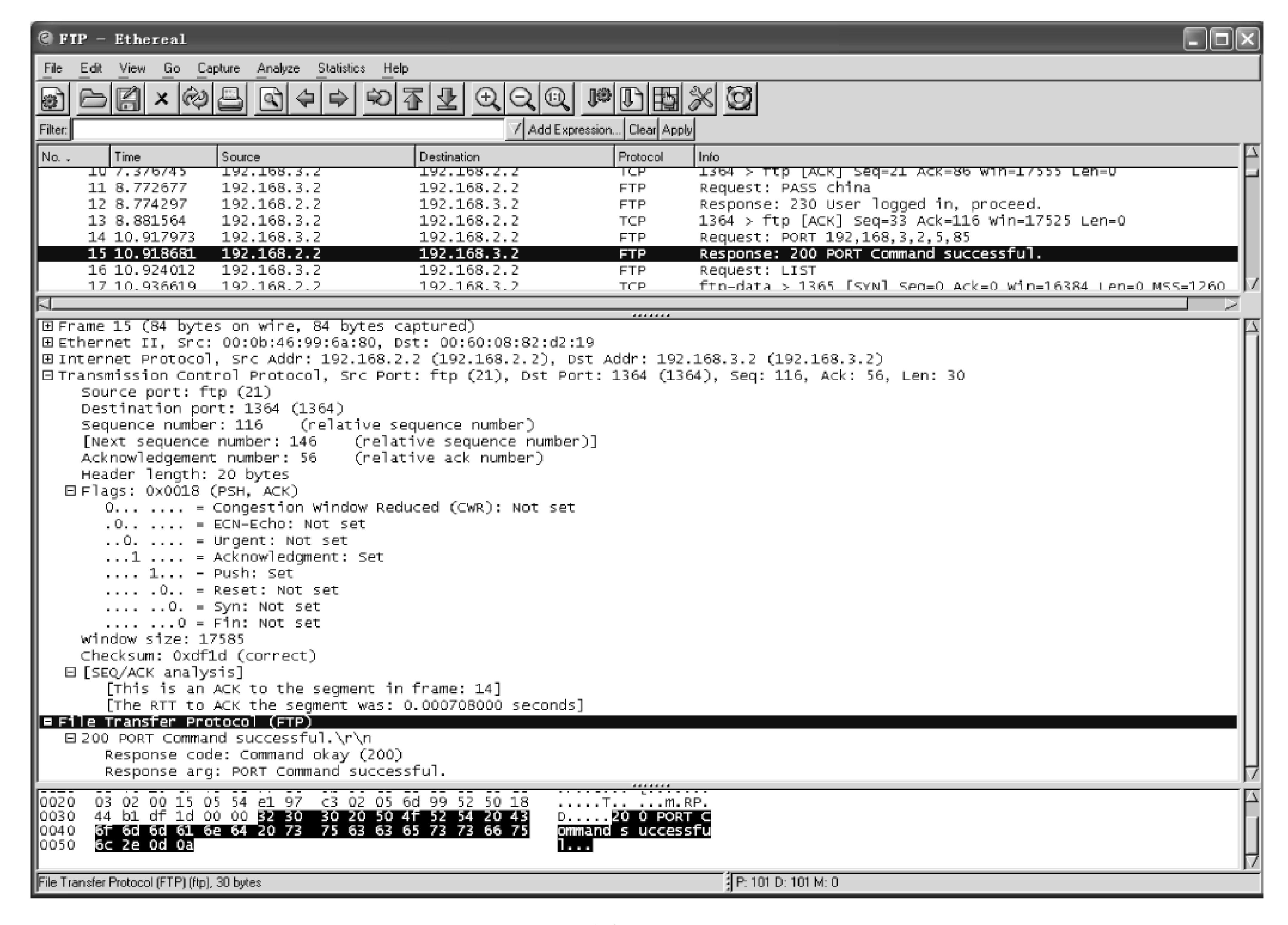

FTP - Ethereal
File Edit View Go Capture Analyze Statistics Help
Filter: Add Expression... Clear Apply
No. . Time Source Destination Protocol Info
11 8.772677 192.168.3.2 192.168.2.2 FTP Request: PASS china
12 8.774297 192.168.2.2 192.168.3.2 FTP Response: 230 User logged in, proceed.
13 8.881564 192.168.3.2 192.168.2.2 TCP 1364 > ftp [ACK] Seq=33 Ack=116 Win=17525 Len=0
14 10.917973 192.168.3.2 192.168.2.2 FTP Request: PORT 192,168,3,2,5,85
15 10.918681 192.168.2.2 192.168.3.2 FTP Response: 200 PORT Command successful.
16 10.924012 192.168.3.2 192.168.2.2 FTP Request: LIST
Frame 15 (84 bytes on wire, 84 bytes captured)
Ethernet II, Src: 00:0b:46:99:6a:80, Dst: 00:60:08:82:d2:19
Internet Protocol, Src Addr: 192.168.2.2 (192.168.2.2), Dst Addr: 192.168.3.2 (192.168.3.2)
Transmission Control Protocol, Src Port: ftp (21), Dst Port: 1364 (1364), Seq: 116, Ack: 56, Len: 30
Source port: ftp (21)
Destination port: 1364 (1364)
Sequence number: 116 (relative sequence number)
[Next sequence number: 146 (relative sequence number)]
Acknowledgement number: 56 (relative ack number)
Header length: 20 bytes
Flags: 0x0018 (PSH, ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 1... = Push: Set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 17585
Checksum: 0xdf1d (correct)
[SEQ/ACK analysis]
[This is an ACK to the segment in frame: 14]
[The RTT to ACK the segment was: 0.000708000 seconds]
File Transfer Protocol (FTP)
200 PORT Command successful.\r\n
Response code: Command okay (200)
Response arg: PORT Command successful.
0020 03 02 00 15 05 54 e1 97 c3 02 05 6d 99 52 50 18 .....T.. ...m.RP.
0030 44 b1 df 1d 00 00 32 30 30 20 50 4f 52 54 20 43 D.....20 0 PORT C
0040 6f 6d 6d 61 6e 64 20 73 75 63 63 65 73 73 66 75 ommand s uccessfu
0050 6c 2e 0d 0a l...
File Transfer Protocol (FTP) (ftp), 30 bytes
P: 101 D: 101 M: 0

图 8-11

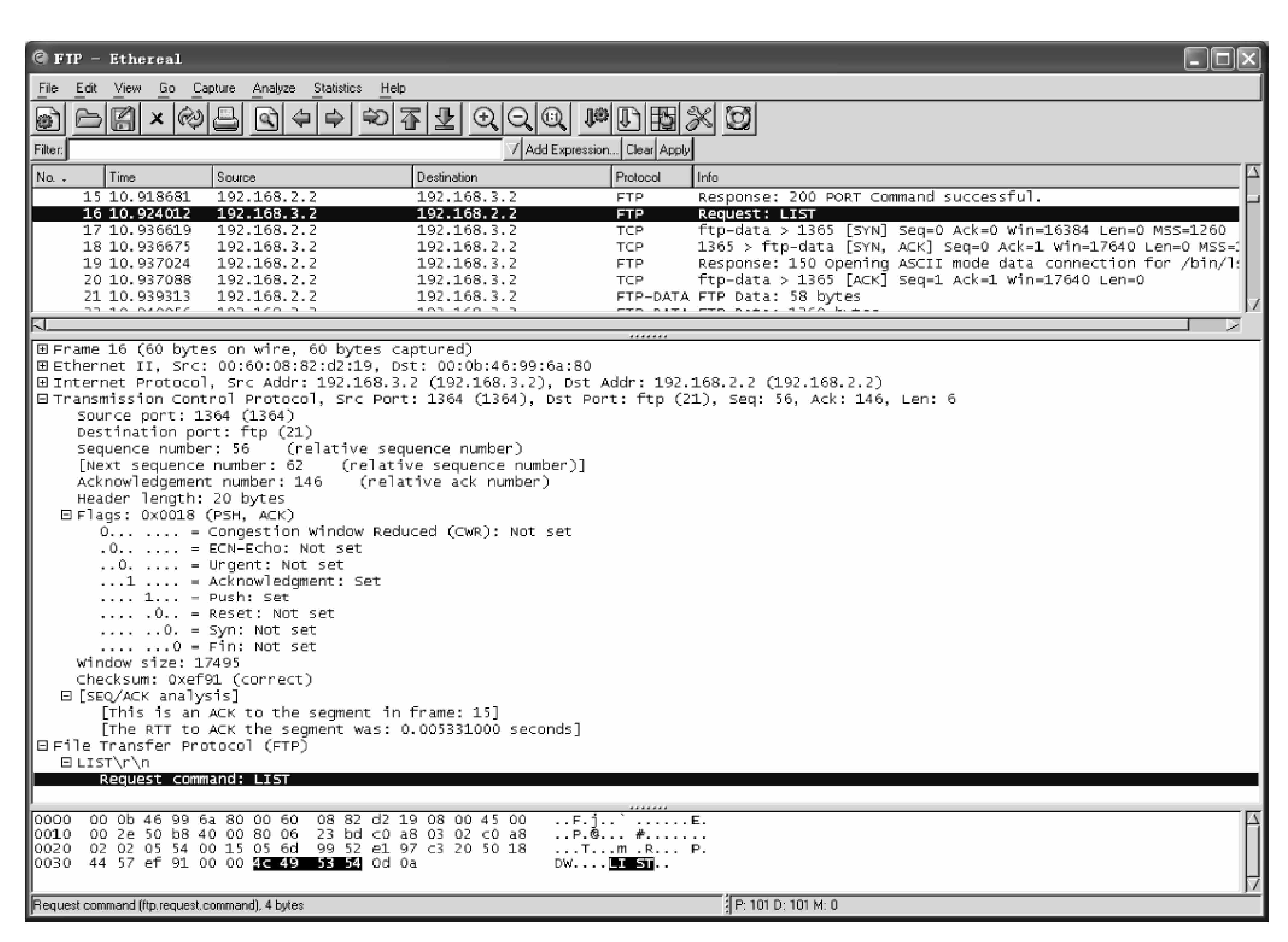

FTP - Ethereal
File Edit View Go Capture Analyze Statistics Help
Filter: Add Expression... Clear Apply
No. . Time Source Destination Protocol Info
15 10.918681 192.168.2.2 192.168.3.2 FTP Response: 200 PORT Command successful.
16 10.924012 192.168.3.2 192.168.2.2 FTP Request: LIST
17 10.936619 192.168.2.2 192.168.3.2 TCP ftp-data > 1365 [SYN] Seq=0 Ack=0 Win=16384 Len=0 MSS=1260
18 10.936675 192.168.3.2 192.168.2.2 TCP 1365 > ftp-data [SYN, ACK] Seq=0 Ack=1 Win=17640 Len=0 MSS=
19 10.937024 192.168.2.2 192.168.3.2 FTP Response: 150 Opening ASCII mode data connection for /bin/l
20 10.937088 192.168.2.2 192.168.3.2 TCP ftp-data > 1365 [ACK] Seq=1 Ack=1 Win=17640 Len=0
21 10.939313 192.168.2.2 192.168.3.2 FTP-DATA FTP Data: 58 bytes
Frame 16 (60 bytes on wire, 60 bytes captured)
Ethernet II, Src: 00:60:08:82:d2:19, Dst: 00:0b:46:99:6a:80
Internet Protocol, Src Addr: 192.168.3.2 (192.168.3.2), Dst Addr: 192.168.2.2 (192.168.2.2)
Transmission Control Protocol, Src Port: 1364 (1364), Dst Port: ftp (21), Seq: 56, Ack: 146, Len: 6
Source port: 1364 (1364)
Destination port: ftp (21)
Sequence number: 56 (relative sequence number)
[Next sequence number: 62 (relative sequence number)]
Acknowledgement number: 146 (relative ack number)
Header length: 20 bytes
Flags: 0x0018 (PSH, ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 1... = Push: Set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 17495
Checksum: 0xef91 (correct)
[SEQ/ACK analysis]
[This is an ACK to the segment in frame: 15]
[The RTT to ACK the segment was: 0.005331000 seconds]
File Transfer Protocol (FTP)
LIST\r\n
Request command: LIST
0000 00 0b 46 99 6a 80 00 60 08 82 d2 19 08 00 45 00 ..F.j..` ......E.
0010 00 2e 50 b8 40 00 80 06 23 bd c0 a8 03 02 c0 a8 ..P.@... #.......
0020 02 02 05 54 00 15 05 6d 99 52 e1 97 c3 20 50 18 ...T...m .R... P.
0030 44 57 ef 91 00 00 4c 49 53 54 0d 0a DW....LI ST..
Request command (ftp.request.command), 4 bytes
P: 101 D: 101 M: 0

图 8-12

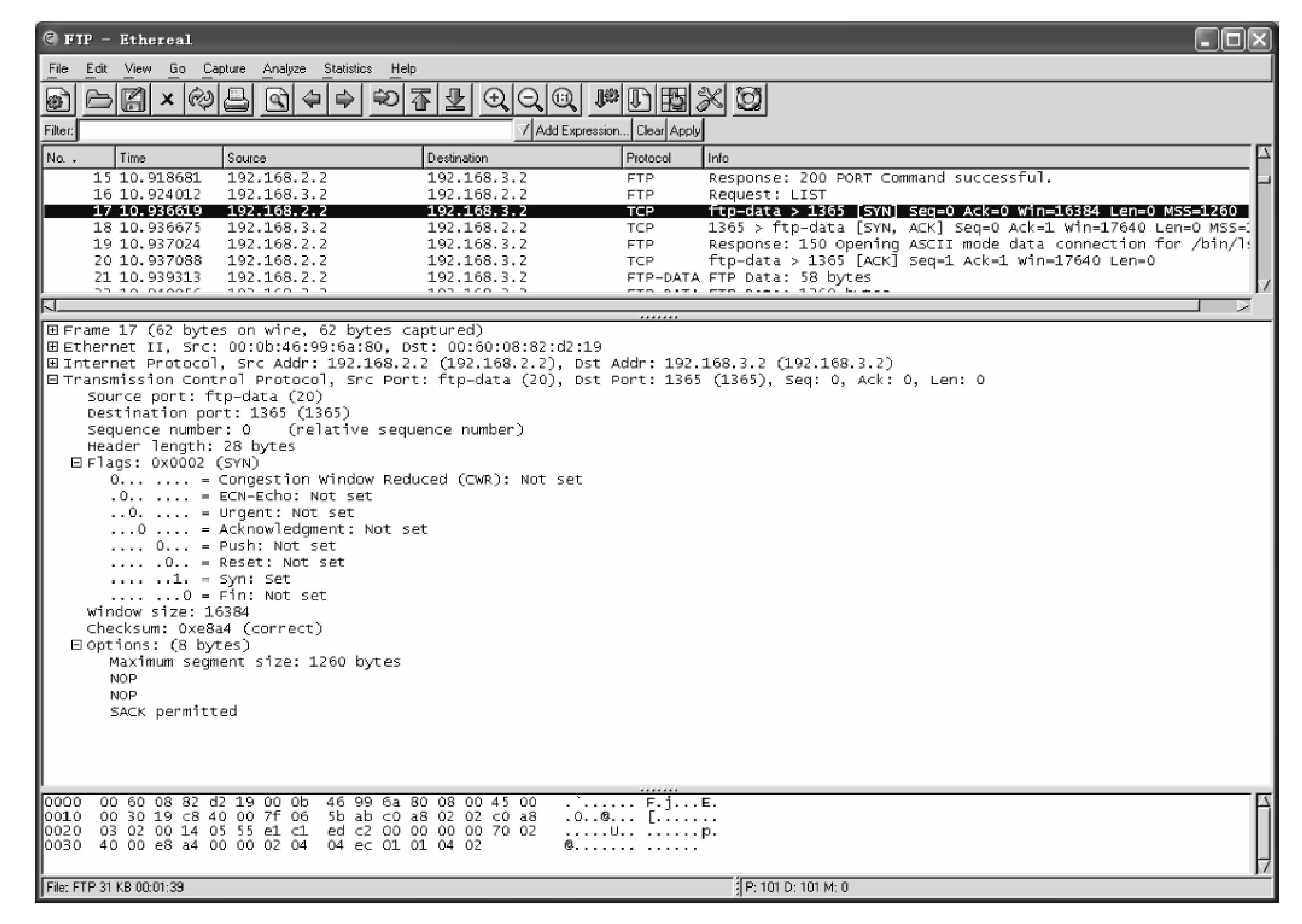

FTP - Ethereal
File Edit View Go Capture Analyze Statistics Help
Filter: Add Expression... Clear Apply
No. Time Source Destination Protocol Info
15 10.918681 192.168.2.2 192.168.3.2 FTP Response: 200 PORT Command successful.
16 10.924012 192.168.3.2 192.168.2.2 FTP Request: LIST
17 10.936619 192.168.2.2 192.168.3.2 TCP ftp-data > 1365 [SYN] Seq=0 Ack=0 Win=16384 Len=0 MSS=1260
18 10.936675 192.168.3.2 192.168.2.2 TCP 1365 > ftp-data [SYN, ACK] Seq=0 Ack=1 Win=17640 Len=0 MSS=
19 10.937024 192.168.2.2 192.168.3.2 FTP Response: 150 Opening ASCII mode data connection for /bin/ls
20 10.937088 192.168.2.2 192.168.3.2 TCP ftp-data > 1365 [ACK] Seq=1 Ack=1 Win=17640 Len=0
21 10.939313 192.168.2.2 192.168.3.2 FTP-DATA FTP Data: 58 bytes
Frame 17 (62 bytes on wire, 62 bytes captured)
Ethernet II, Src: 00:0b:46:99:6a:80, Dst: 00:60:08:82:d2:19
Internet Protocol, Src Addr: 192.168.2.2 (192.168.2.2), Dst Addr: 192.168.3.2 (192.168.3.2)
Transmission Control Protocol, Src Port: ftp-data (20), Dst Port: 1365 (1365), Seq: 0, Ack: 0, Len: 0
Source port: ftp-data (20)
Destination port: 1365 (1365)
Sequence number: 0 (relative sequence number)
Header length: 28 bytes
Flags: 0x0002 (SYN)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...0 .... = Acknowledgment: Not set
.... 0... = Push: Not set
.... .0.. = Reset: Not set
.... ..1. = Syn: Set
.... ...0 = Fin: Not set
Window size: 16384
Checksum: 0xe8a4 (correct)
Options: (8 bytes)
Maximum segment size: 1260 bytes
NOP
NOP
SACK permitted
0000 00 60 08 82 d2 19 00 0b 46 99 6a 80 08 00 45 00
0010 00 30 19 c8 40 00 7f 06 5b ab c0 a8 02 02 c0 a8
0020 03 02 00 14 05 55 e1 c1 ed c2 00 00 00 00 70 02
0030 40 00 e8 a4 00 00 02 04 04 ec 01 01 04 02
File: FTP 31 KB 00:01:39
P: 101 D: 101 M: 0

图　8-13

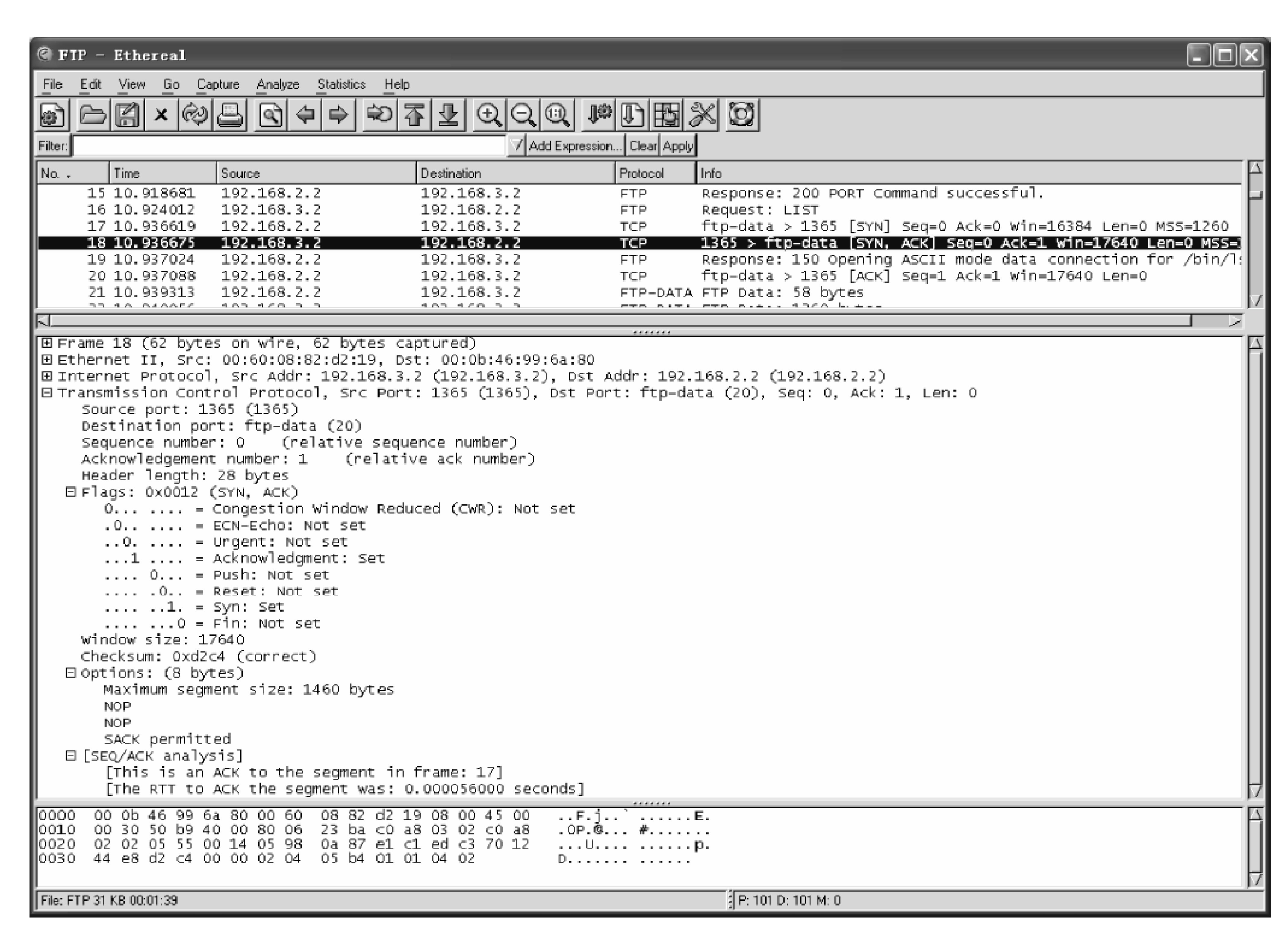

FTP - Ethereal
File Edit View Go Capture Analyze Statistics Help
Filter: Add Expression... Clear Apply
No. Time Source Destination Protocol Info
15 10.918681 192.168.2.2 192.168.3.2 FTP Response: 200 PORT Command successful.
16 10.924012 192.168.3.2 192.168.2.2 FTP Request: LIST
17 10.936619 192.168.2.2 192.168.3.2 TCP ftp-data > 1365 [SYN] Seq=0 Ack=0 Win=16384 Len=0 MSS=1260
18 10.936675 192.168.3.2 192.168.2.2 TCP 1365 > ftp-data [SYN, ACK] Seq=0 Ack=1 Win=17640 Len=0 MSS=
19 10.937024 192.168.2.2 192.168.3.2 FTP Response: 150 Opening ASCII mode data connection for /bin/ls
20 10.937088 192.168.2.2 192.168.3.2 TCP ftp-data > 1365 [ACK] Seq=1 Ack=1 Win=17640 Len=0
21 10.939313 192.168.2.2 192.168.3.2 FTP-DATA FTP Data: 58 bytes
Frame 18 (62 bytes on wire, 62 bytes captured)
Ethernet II, Src: 00:60:08:82:d2:19, Dst: 00:0b:46:99:6a:80
Internet Protocol, Src Addr: 192.168.3.2 (192.168.3.2), Dst Addr: 192.168.2.2 (192.168.2.2)
Transmission Control Protocol, Src Port: 1365 (1365), Dst Port: ftp-data (20), Seq: 0, Ack: 1, Len: 0
Source port: 1365 (1365)
Destination port: ftp-data (20)
Sequence number: 0 (relative sequence number)
Acknowledgement number: 1 (relative ack number)
Header length: 28 bytes
Flags: 0x0012 (SYN, ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 0... = Push: Not set
.... .0.. = Reset: Not set
.... ..1. = Syn: Set
.... ...0 = Fin: Not set
Window size: 17640
Checksum: 0xd2c4 (correct)
Options: (8 bytes)
Maximum segment size: 1460 bytes
NOP
NOP
SACK permitted
[SEQ/ACK analysis]
[This is an ACK to the segment in frame: 17]
[The RTT to ACK the segment was: 0.000056000 seconds]
0000 00 0b 46 99 6a 80 00 60 08 82 d2 19 08 00 45 00
0010 00 30 50 b9 40 00 80 06 23 ba c0 a8 03 02 c0 a8
0020 02 02 05 55 00 14 05 98 0a 87 e1 c1 ed c3 70 12
0030 44 e8 d2 c4 00 00 02 04 05 b4 01 01 04 02
File: FTP 31 KB 00:01:39
P: 101 D: 101 M: 0

图　8-14

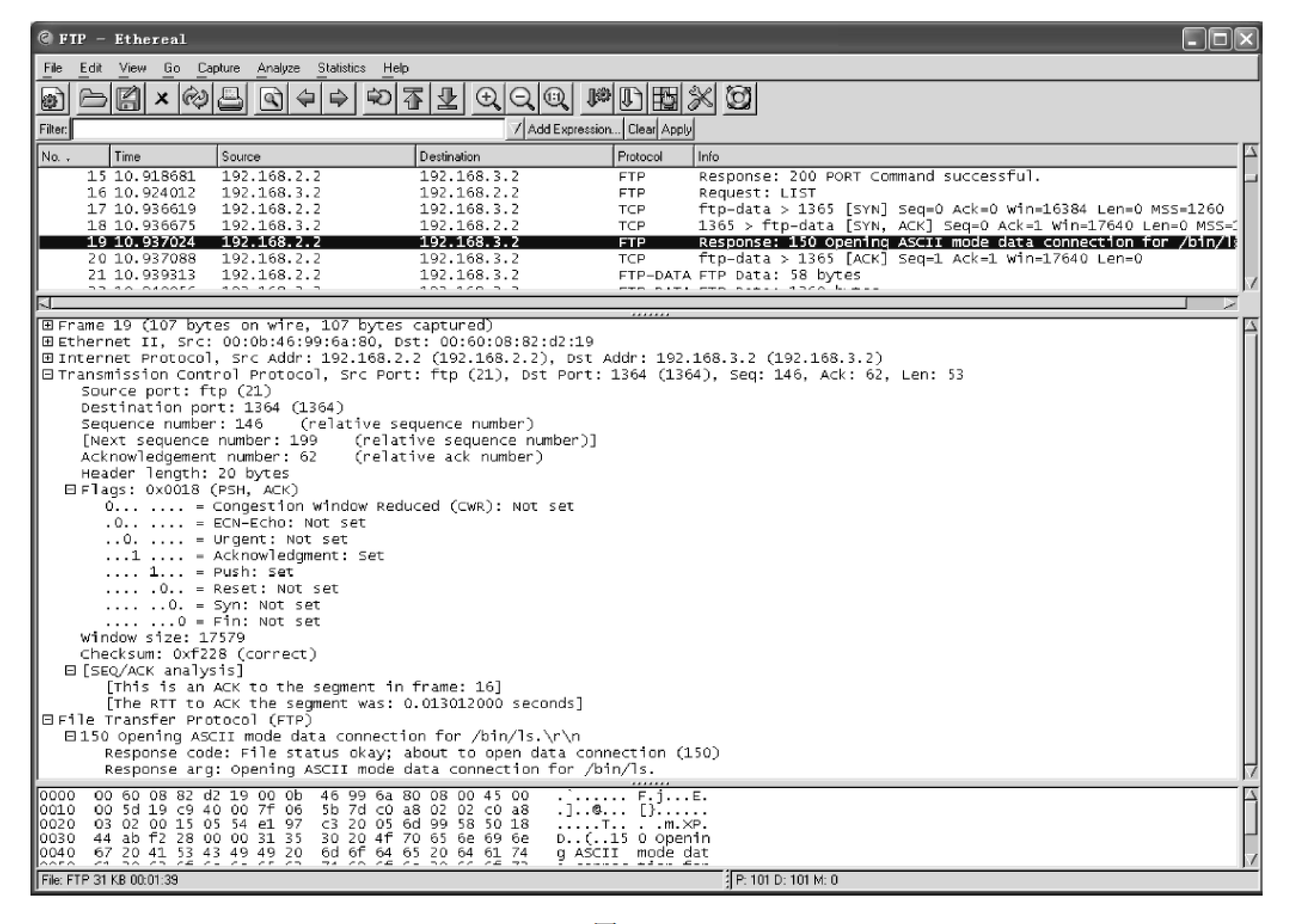

FTP - Ethereal
File Edit View Go Capture Analyze Statistics Help
Filter:
Add Expression... Clear Apply
No. Time Source Destination Protocol Info
15 10.918681 192.168.2.2 192.168.3.2 FTP Response: 200 PORT Command successful.
16 10.924012 192.168.3.2 192.168.2.2 FTP Request: LIST
17 10.936619 192.168.2.2 192.168.3.2 TCP ftp-data > 1365 [SYN] Seq=0 Ack=0 Win=16384 Len=0 MSS=1260
18 10.936675 192.168.3.2 192.168.2.2 TCP 1365 > ftp-data [SYN, ACK] Seq=0 Ack=1 Win=17640 Len=0 MSS=
19 10.937024 192.168.2.2 192.168.3.2 FTP Response: 150 Opening ASCII mode data connection for /bin/l
20 10.937088 192.168.2.2 192.168.3.2 TCP ftp-data > 1365 [ACK] Seq=1 Ack=1 Win=17640 Len=0
21 10.939313 192.168.2.2 192.168.3.2 FTP-DATA FTP Data: 58 bytes
Frame 19 (107 bytes on wire, 107 bytes captured)
Ethernet II, Src: 00:0b:46:99:6a:80, Dst: 00:60:08:82:d2:19
Internet Protocol, Src Addr: 192.168.2.2 (192.168.2.2), Dst Addr: 192.168.3.2 (192.168.3.2)
Transmission Control Protocol, Src Port: ftp (21), Dst Port: 1364 (1364), Seq: 146, Ack: 62, Len: 53
Source port: ftp (21)
Destination port: 1364 (1364)
Sequence number: 146 (relative sequence number)
[Next sequence number: 199 (relative sequence number)]
Acknowledgement number: 62 (relative ack number)
Header length: 20 bytes
Flags: 0x0018 (PSH, ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 1... = Push: Set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 17579
Checksum: 0xf228 (correct)
[SEQ/ACK analysis]
[This is an ACK to the segment in frame: 16]
[The RTT to ACK the segment was: 0.013012000 seconds]
File Transfer Protocol (FTP)
150 Opening ASCII mode data connection for /bin/ls.\r\n
Response code: File status okay; about to open data connection (150)
Response arg: Opening ASCII mode data connection for /bin/ls.
0000 00 60 08 82 d2 19 00 0b 46 99 6a 80 08 00 45 00 .`...... F.j...E.
0010 00 5d 19 c9 40 00 7f 06 5b 7d c0 a8 02 02 c0 a8 .]..@... [}......
0020 03 02 00 15 05 54 e1 97 c3 20 05 6d 99 58 50 18 .....T.. . .m.XP.
0030 44 ab f2 28 00 00 31 35 30 20 4f 70 65 6e 69 6e D..(..15 0 Openin
0040 67 20 41 53 43 49 49 20 6d 6f 64 65 20 64 61 74 g ASCII mode dat
File: FTP 31 KB 00:01:39
P: 101 D: 101 M: 0

图 8-15

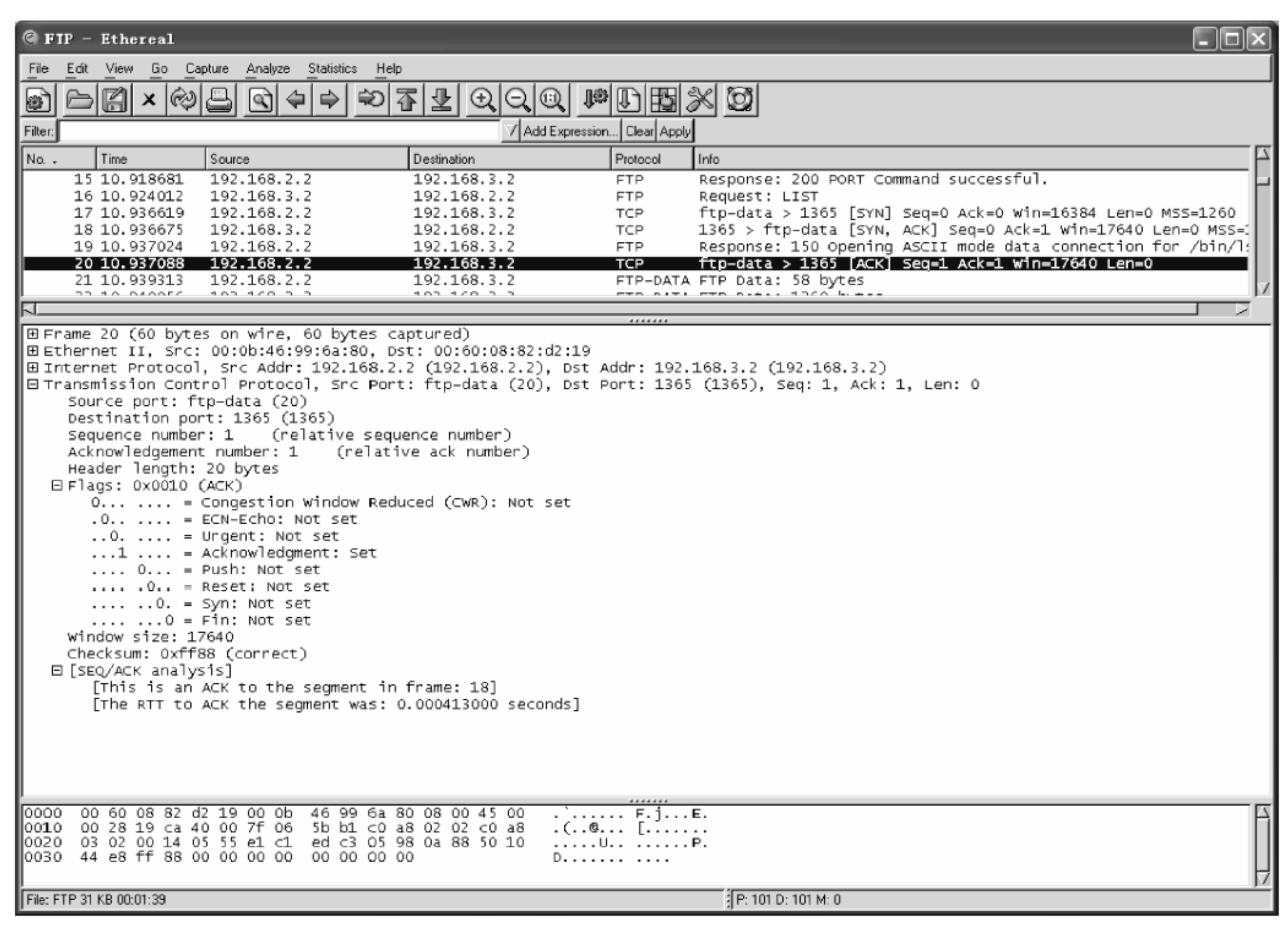

FTP - Ethereal
File Edit View Go Capture Analyze Statistics Help
Filter:
Add Expression... Clear Apply
No. Time Source Destination Protocol Info
15 10.918681 192.168.2.2 192.168.3.2 FTP Response: 200 PORT Command successful.
16 10.924012 192.168.3.2 192.168.2.2 FTP Request: LIST
17 10.936619 192.168.2.2 192.168.3.2 TCP ftp-data > 1365 [SYN] Seq=0 Ack=0 Win=16384 Len=0 MSS=1260
18 10.936675 192.168.3.2 192.168.2.2 TCP 1365 > ftp-data [SYN, ACK] Seq=0 Ack=1 Win=17640 Len=0 MSS=
19 10.937024 192.168.2.2 192.168.3.2 FTP Response: 150 Opening ASCII mode data connection for /bin/l
20 10.937088 192.168.2.2 192.168.3.2 TCP ftp-data > 1365 [ACK] Seq=1 Ack=1 Win=17640 Len=0
21 10.939313 192.168.2.2 192.168.3.2 FTP-DATA FTP Data: 58 bytes
Frame 20 (60 bytes on wire, 60 bytes captured)
Ethernet II, Src: 00:0b:46:99:6a:80, Dst: 00:60:08:82:d2:19
Internet Protocol, Src Addr: 192.168.2.2 (192.168.2.2), Dst Addr: 192.168.3.2 (192.168.3.2)
Transmission Control Protocol, Src Port: ftp-data (20), Dst Port: 1365 (1365), Seq: 1, Ack: 1, Len: 0
Source port: ftp-data (20)
Destination port: 1365 (1365)
Sequence number: 1 (relative sequence number)
Acknowledgement number: 1 (relative ack number)
Header length: 20 bytes
Flags: 0x0010 (ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 0... = Push: Not set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 17640
Checksum: 0xff88 (correct)
[SEQ/ACK analysis]
[This is an ACK to the segment in frame: 18]
[The RTT to ACK the segment was: 0.000413000 seconds]
0000 00 60 08 82 d2 19 00 0b 46 99 6a 80 08 00 45 00 .`...... F.j...E.
0010 00 28 19 ca 40 00 7f 06 5b b1 c0 a8 02 02 c0 a8 .(..@... [.......
0020 03 02 00 14 05 55 e1 c1 ed c3 05 98 0a 88 50 10 .....U.. ......P.
0030 44 e8 ff 88 00 00 00 00 00 00 00 00 D....... ....
File: FTP 31 KB 00:01:39
P: 101 D: 101 M: 0

图 8-16

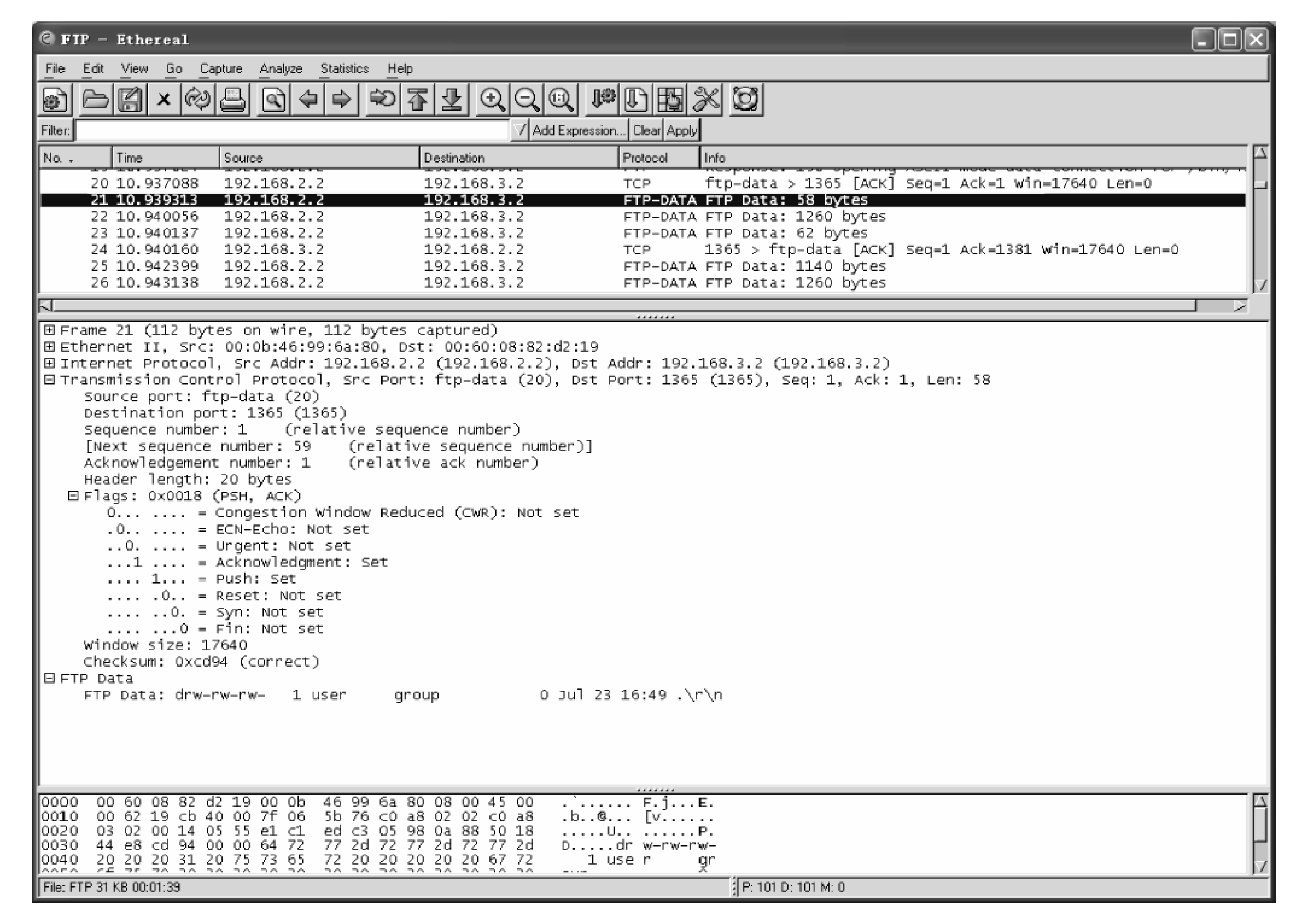

FTP - Ethereal
File Edit View Go Capture Analyze Statistics Help
Filter:
Add Expression... Clear Apply
No. Time Source Destination Protocol Info
20 10.937088 192.168.2.2 192.168.3.2 TCP ftp-data > 1365 [ACK] Seq=1 Ack=1 Win=17640 Len=0
21 10.939313 192.168.2.2 192.168.3.2 FTP-DATA FTP Data: 58 bytes
22 10.940056 192.168.2.2 192.168.3.2 FTP-DATA FTP Data: 1260 bytes
23 10.940137 192.168.2.2 192.168.3.2 FTP-DATA FTP Data: 62 bytes
24 10.940160 192.168.3.2 192.168.2.2 TCP 1365 > ftp-data [ACK] Seq=1 Ack=1381 Win=17640 Len=0
25 10.942399 192.168.2.2 192.168.3.2 FTP-DATA FTP Data: 1140 bytes
26 10.943138 192.168.2.2 192.168.3.2 FTP-DATA FTP Data: 1260 bytes
Frame 21 (112 bytes on wire, 112 bytes captured)
Ethernet II, Src: 00:0b:46:99:6a:80, Dst: 00:60:08:82:d2:19
Internet Protocol, Src Addr: 192.168.2.2 (192.168.2.2), Dst Addr: 192.168.3.2 (192.168.3.2)
Transmission Control Protocol, Src Port: ftp-data (20), Dst Port: 1365 (1365), Seq: 1, Ack: 1, Len: 58
Source port: ftp-data (20)
Destination port: 1365 (1365)
Sequence number: 1 (relative sequence number)
[Next sequence number: 59 (relative sequence number)]
Acknowledgement number: 1 (relative ack number)
Header length: 20 bytes
Flags: 0x0018 (PSH, ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 1... = Push: Set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 17640
Checksum: 0xcd94 (correct)
FTP Data
FTP Data: drw-rw-rw- 1 user group 0 Jul 23 16:49 .\r\n
0000 00 60 08 82 d2 19 00 0b 46 99 6a 80 08 00 45 00 .`...... F.j...E.
0010 00 62 19 cb 40 00 7f 06 5b 76 c0 a8 02 02 c0 a8 .b..@... [v......
0020 03 02 00 14 05 55 e1 c1 ed c3 05 98 0a 88 50 18 .....U.. ......P.
0030 44 e8 cd 94 00 00 64 72 77 2d 72 77 2d 72 77 2d D.....dr w-rw-rw-
0040 20 20 20 31 20 75 73 65 72 20 20 20 20 20 67 72 1 use r gr
File: FTP 31 KB 00:01:39
P: 101 D: 101 M: 0

图　8-17

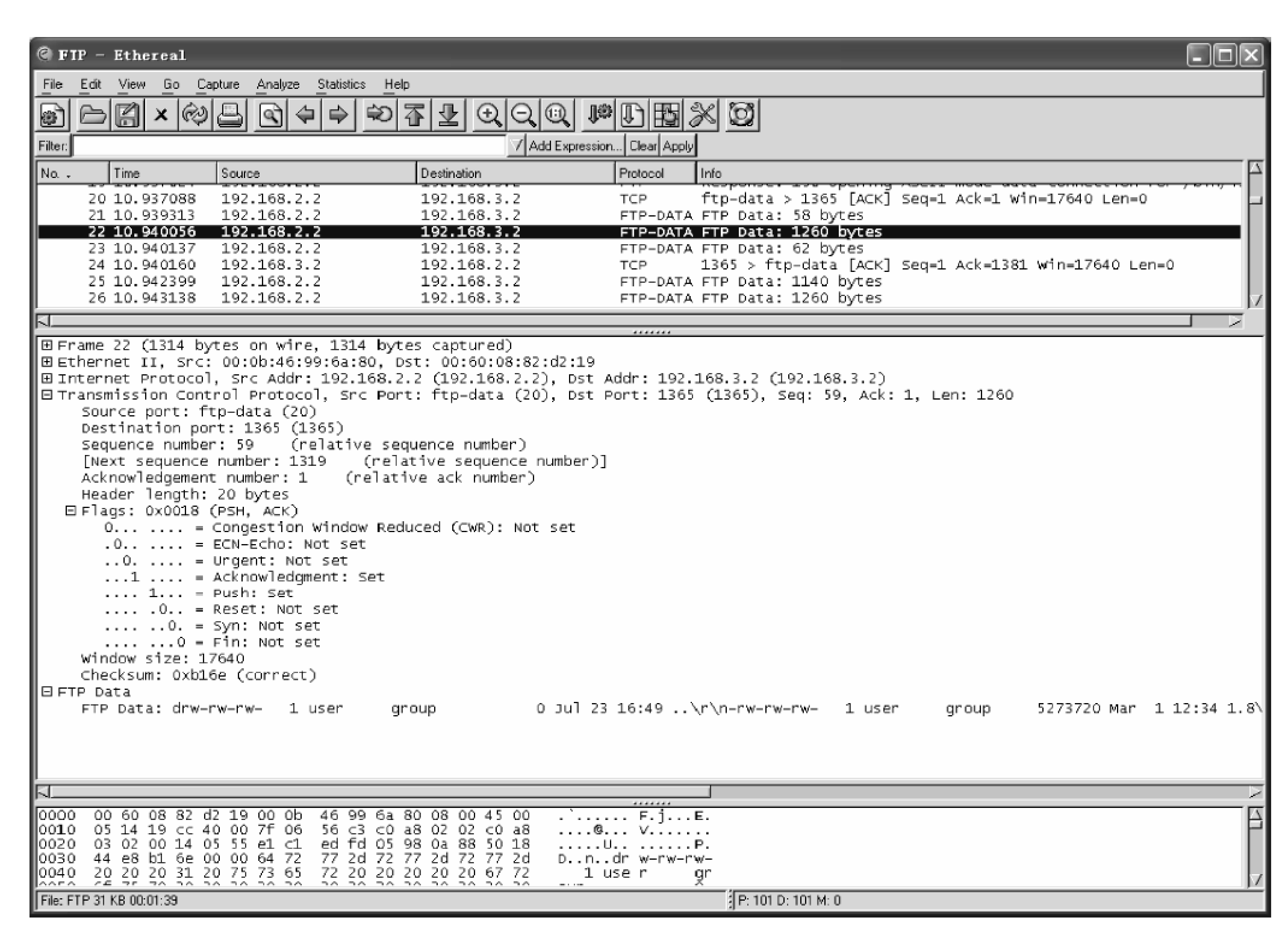

FTP - Ethereal
File Edit View Go Capture Analyze Statistics Help
Filter:
Add Expression... Clear Apply
No. Time Source Destination Protocol Info
20 10.937088 192.168.2.2 192.168.3.2 TCP ftp-data > 1365 [ACK] Seq=1 Ack=1 Win=17640 Len=0
21 10.939313 192.168.2.2 192.168.3.2 FTP-DATA FTP Data: 58 bytes
22 10.940056 192.168.2.2 192.168.3.2 FTP-DATA FTP Data: 1260 bytes
23 10.940137 192.168.2.2 192.168.3.2 FTP-DATA FTP Data: 62 bytes
24 10.940160 192.168.3.2 192.168.2.2 TCP 1365 > ftp-data [ACK] Seq=1 Ack=1381 Win=17640 Len=0
25 10.942399 192.168.2.2 192.168.3.2 FTP-DATA FTP Data: 1140 bytes
26 10.943138 192.168.2.2 192.168.3.2 FTP-DATA FTP Data: 1260 bytes
Frame 22 (1314 bytes on wire, 1314 bytes captured)
Ethernet II, Src: 00:0b:46:99:6a:80, Dst: 00:60:08:82:d2:19
Internet Protocol, Src Addr: 192.168.2.2 (192.168.2.2), Dst Addr: 192.168.3.2 (192.168.3.2)
Transmission Control Protocol, Src Port: ftp-data (20), Dst Port: 1365 (1365), Seq: 59, Ack: 1, Len: 1260
Source port: ftp-data (20)
Destination port: 1365 (1365)
Sequence number: 59 (relative sequence number)
[Next sequence number: 1319 (relative sequence number)]
Acknowledgement number: 1 (relative ack number)
Header length: 20 bytes
Flags: 0x0018 (PSH, ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 1... = Push: Set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 17640
Checksum: 0xb16e (correct)
FTP Data
FTP Data: drw-rw-rw- 1 user group 0 Jul 23 16:49 ..\r\n-rw-rw-rw- 1 user group 5273720 Mar 1 12:34 1.8\
0000 00 60 08 82 d2 19 00 0b 46 99 6a 80 08 00 45 00 .`...... F.j...E.
0010 05 14 19 cc 40 00 7f 06 56 c3 c0 a8 02 02 c0 a8 ....@... V.......
0020 03 02 00 14 05 55 e1 c1 ed fd 05 98 0a 88 50 18 .....U.. ......P.
0030 44 e8 b1 6e 00 00 64 72 77 2d 72 77 2d 72 77 2d D..n..dr w-rw-rw-
0040 20 20 20 31 20 75 73 65 72 20 20 20 20 20 67 72 1 use r gr
File: FTP 31 KB 00:01:39
P: 101 D: 101 M: 0

图　8-18

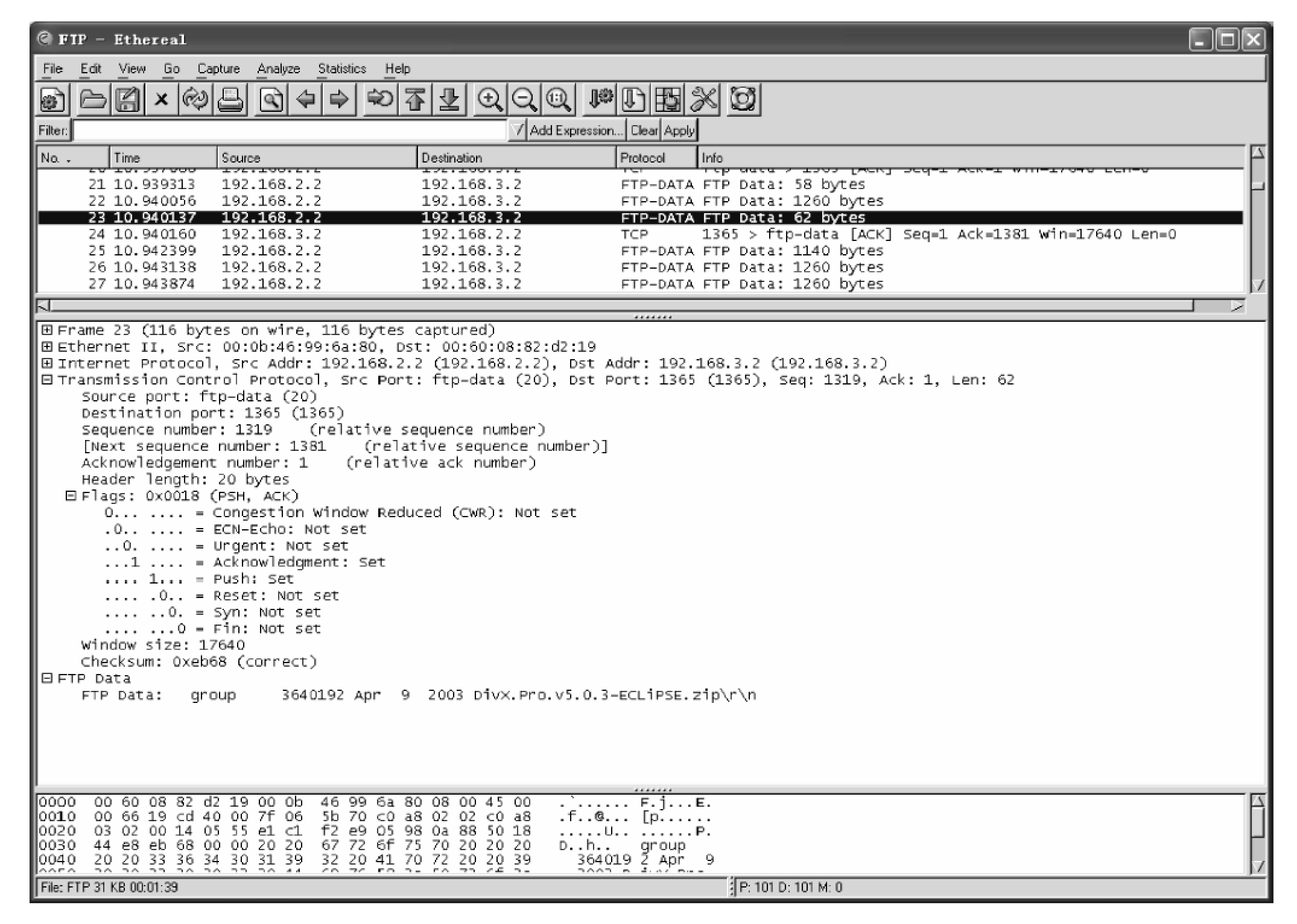

FTP - Ethereal
File Edit View Go Capture Analyze Statistics Help
Filter: Add Expression... Clear Apply
No. Time Source Destination Protocol Info
21 10.939313 192.168.2.2 192.168.3.2 FTP-DATA FTP Data: 58 bytes
22 10.940056 192.168.2.2 192.168.3.2 FTP-DATA FTP Data: 1260 bytes
23 10.940137 192.168.2.2 192.168.3.2 FTP-DATA FTP Data: 62 bytes
24 10.940160 192.168.3.2 192.168.2.2 TCP 1365 > ftp-data [ACK] Seq=1 Ack=1381 Win=17640 Len=0
25 10.942399 192.168.2.2 192.168.3.2 FTP-DATA FTP Data: 1140 bytes
26 10.943138 192.168.2.2 192.168.3.2 FTP-DATA FTP Data: 1260 bytes
27 10.943874 192.168.2.2 192.168.3.2 FTP-DATA FTP Data: 1260 bytes
Frame 23 (116 bytes on wire, 116 bytes captured)
Ethernet II, Src: 00:0b:46:99:6a:80, Dst: 00:60:08:82:d2:19
Internet Protocol, Src Addr: 192.168.2.2 (192.168.2.2), Dst Addr: 192.168.3.2 (192.168.3.2)
Transmission Control Protocol, Src Port: ftp-data (20), Dst Port: 1365 (1365), Seq: 1319, Ack: 1, Len: 62
Source port: ftp-data (20)
Destination port: 1365 (1365)
Sequence number: 1319 (relative sequence number)
[Next sequence number: 1381 (relative sequence number)]
Acknowledgement number: 1 (relative ack number)
Header length: 20 bytes
Flags: 0x0018 (PSH, ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 1... = Push: Set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 17640
Checksum: 0xeb68 (correct)
FTP Data
FTP Data: group 3640192 Apr 9 2003 DivX.Pro.v5.0.3-ECLiPSE.zip\r\n
0000 00 60 08 82 d2 19 00 0b 46 99 6a 80 08 00 45 00 .`...... F.j...E.
0010 00 66 19 cd 40 00 7f 06 5b 70 c0 a8 02 02 c0 a8 .f..@... [p......
0020 03 02 00 14 05 55 e1 c1 f2 e9 05 98 0a 88 50 18 .....U.. ......P.
0030 44 e8 eb 68 00 00 20 20 67 72 6f 75 70 20 20 20 D..h.. group
0040 20 20 33 36 34 30 31 39 32 20 41 70 72 20 20 39 3640192 Apr 9
File: FTP 31 KB 00:01:39
P: 101 D: 101 M: 0

图 8-19

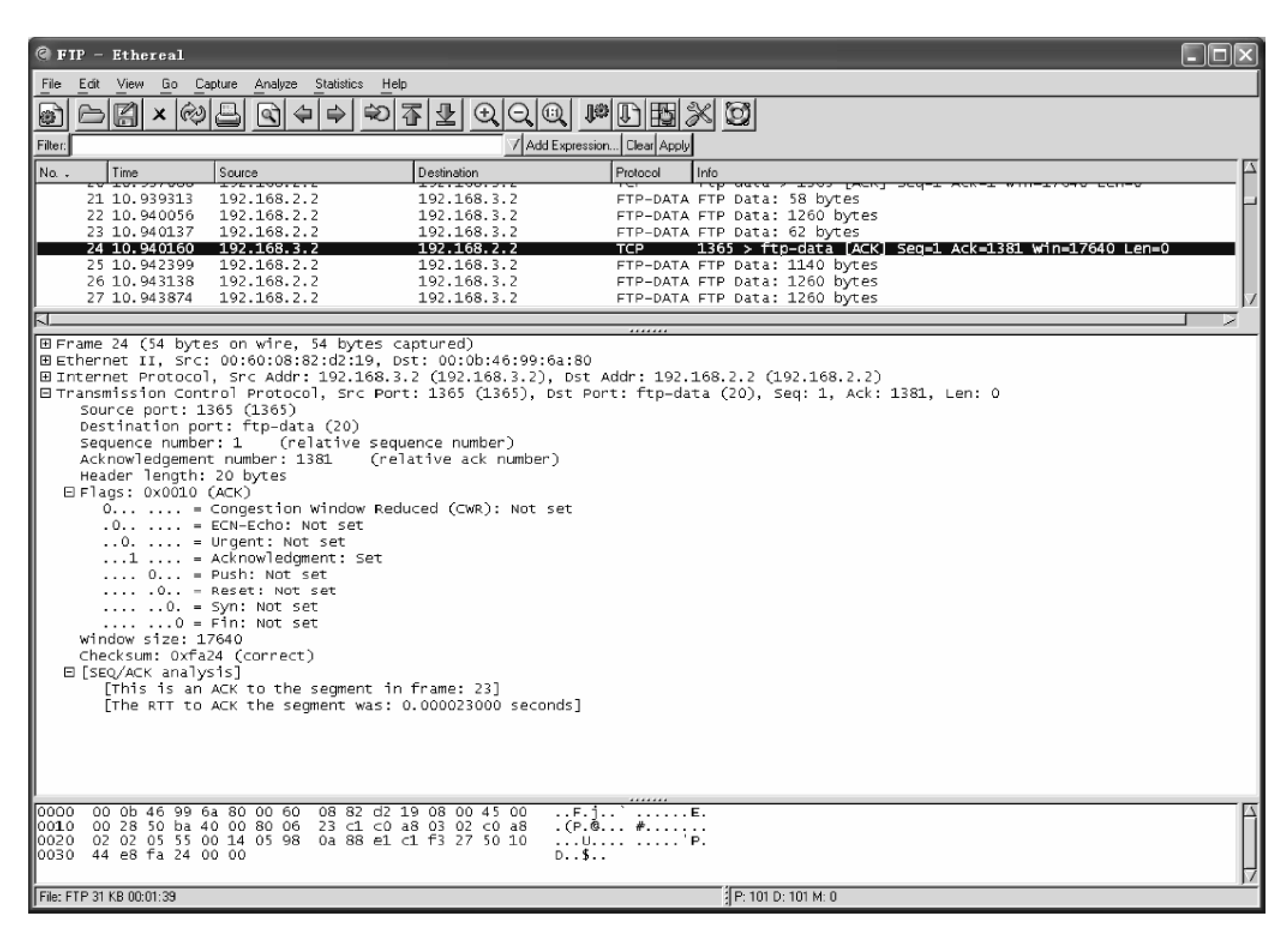

FTP - Ethereal
File Edit View Go Capture Analyze Statistics Help
Filter: Add Expression... Clear Apply
No. Time Source Destination Protocol Info
21 10.939313 192.168.2.2 192.168.3.2 FTP-DATA FTP Data: 58 bytes
22 10.940056 192.168.2.2 192.168.3.2 FTP-DATA FTP Data: 1260 bytes
23 10.940137 192.168.2.2 192.168.3.2 FTP-DATA FTP Data: 62 bytes
24 10.940160 192.168.3.2 192.168.2.2 TCP 1365 > ftp-data [ACK] Seq=1 Ack=1381 Win=17640 Len=0
25 10.942399 192.168.2.2 192.168.3.2 FTP-DATA FTP Data: 1140 bytes
26 10.943138 192.168.2.2 192.168.3.2 FTP-DATA FTP Data: 1260 bytes
27 10.943874 192.168.2.2 192.168.3.2 FTP-DATA FTP Data: 1260 bytes
Frame 24 (54 bytes on wire, 54 bytes captured)
Ethernet II, Src: 00:60:08:82:d2:19, Dst: 00:0b:46:99:6a:80
Internet Protocol, Src Addr: 192.168.3.2 (192.168.3.2), Dst Addr: 192.168.2.2 (192.168.2.2)
Transmission Control Protocol, Src Port: 1365 (1365), Dst Port: ftp-data (20), Seq: 1, Ack: 1381, Len: 0
Source port: 1365 (1365)
Destination port: ftp-data (20)
Sequence number: 1 (relative sequence number)
Acknowledgement number: 1381 (relative ack number)
Header length: 20 bytes
Flags: 0x0010 (ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 0... = Push: Not set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 17640
Checksum: 0xfa24 (correct)
[SEQ/ACK analysis]
[This is an ACK to the segment in frame: 23]
[The RTT to ACK the segment was: 0.000023000 seconds]
0000 00 0b 46 99 6a 80 00 60 08 82 d2 19 08 00 45 00 ..F.j..` ......E.
0010 00 28 50 ba 40 00 80 06 23 c1 c0 a8 03 02 c0 a8 .(P.@... #.......
0020 02 02 05 55 00 14 05 98 0a 88 e1 c1 f3 27 50 10 ...U.... .....'P.
0030 44 e8 fa 24 00 00 D..$..
File: FTP 31 KB 00:01:39
P: 101 D: 101 M: 0

图 8-20

当数据传完后，服务器端和客户端经过 TCP 4 次握手关闭数据连接，保持控制连接，这时，还可以进行其他的数据传输，如图 8-21 所示。接着服务器通过控制连接，告诉客户端数据连接已完成，如图 8-22 所示。

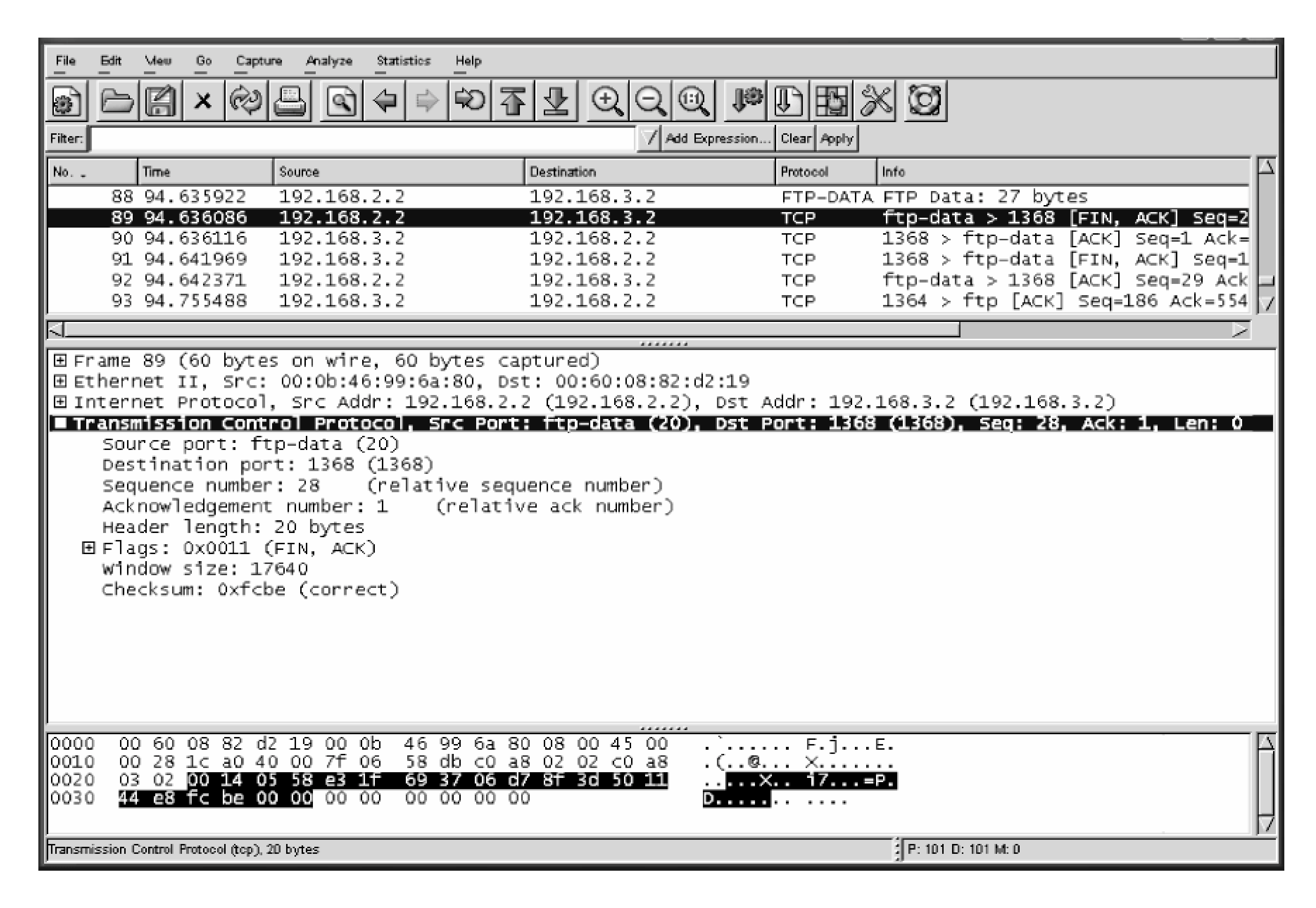



图　8-21

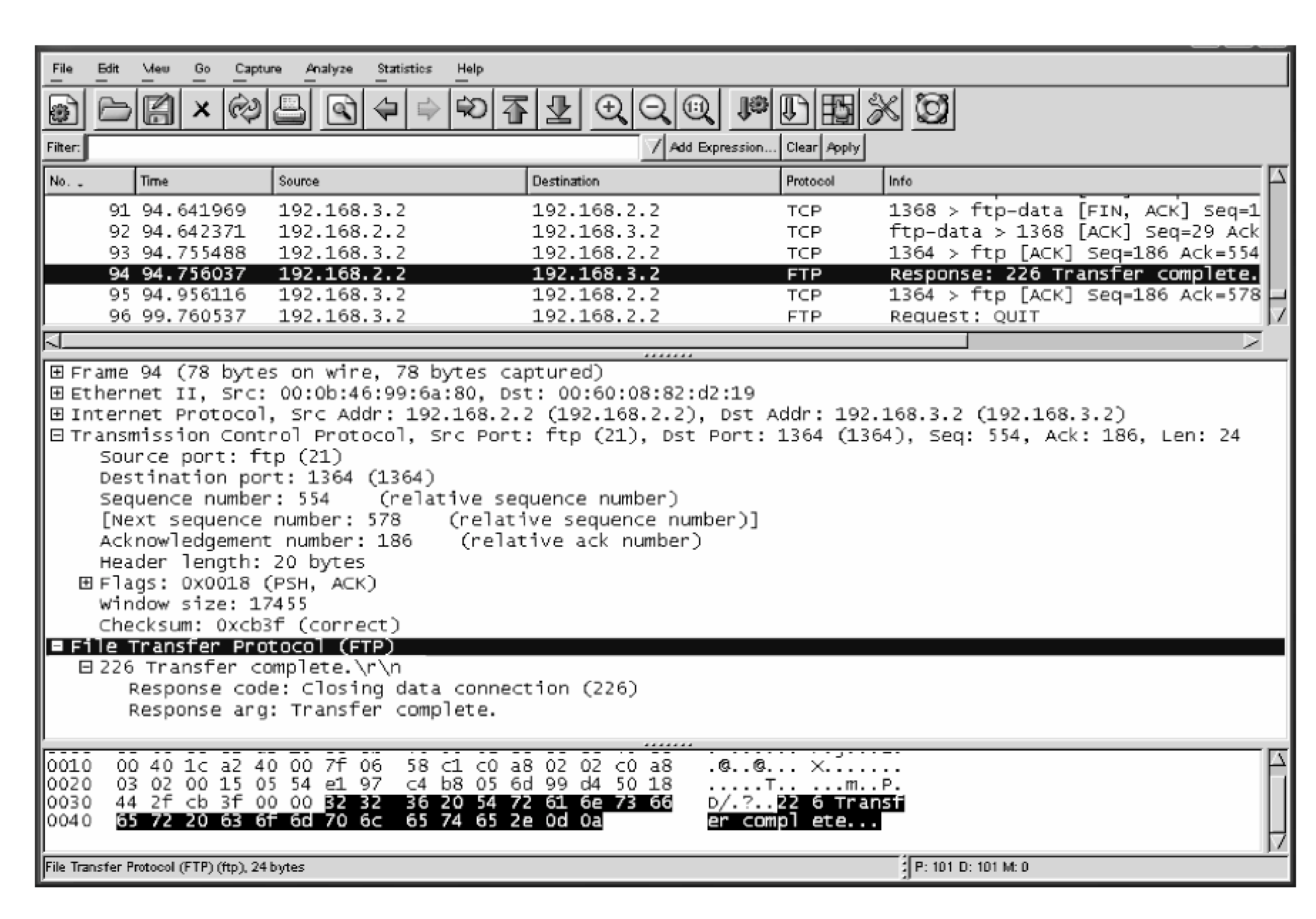



图　8-22

客户端发送 TCP 确认，如图 8-23 所示。

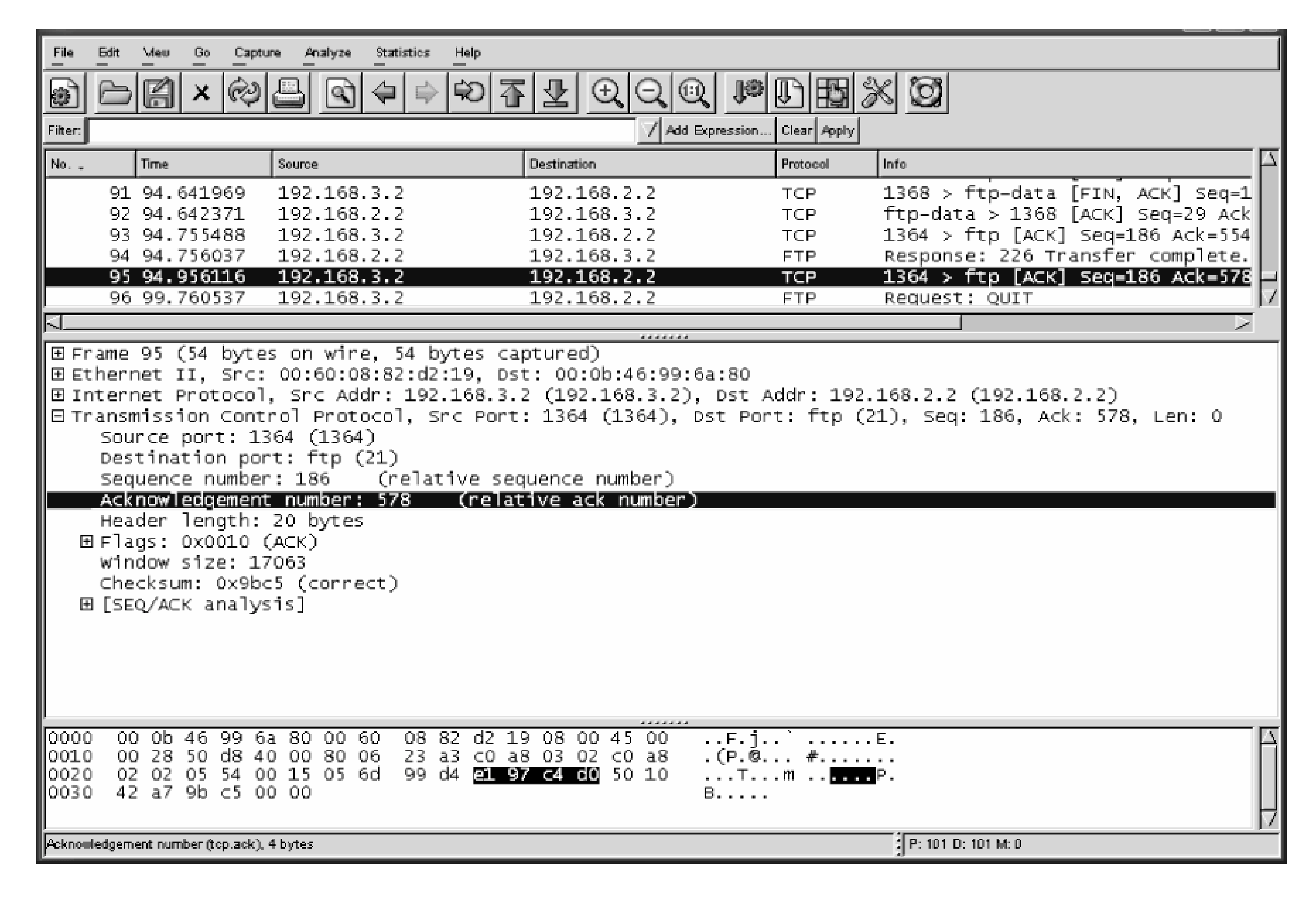



图 8-23

客户端发送退出请求命令 QUIT，如图 8-24 所示。

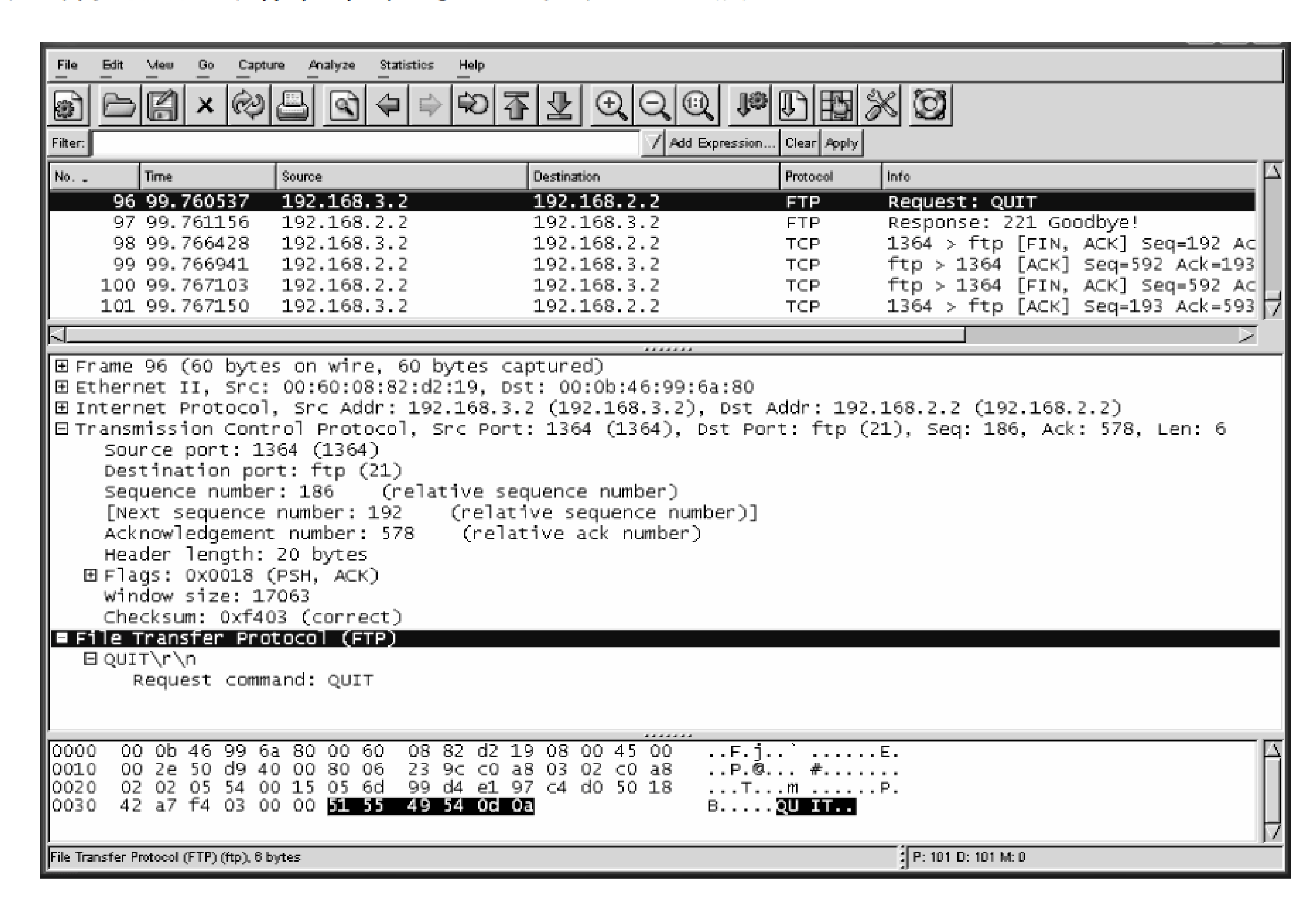



图 8-24

服务器发送 221 回应，告诉客户端将断开控制连接，如图 8-25 所示。

接着就是 TCP 的 4 次握手，结束整个 FTP 会话，如图 8-26 所示。

到此，分析了 FTP 协议及会话的全过程。

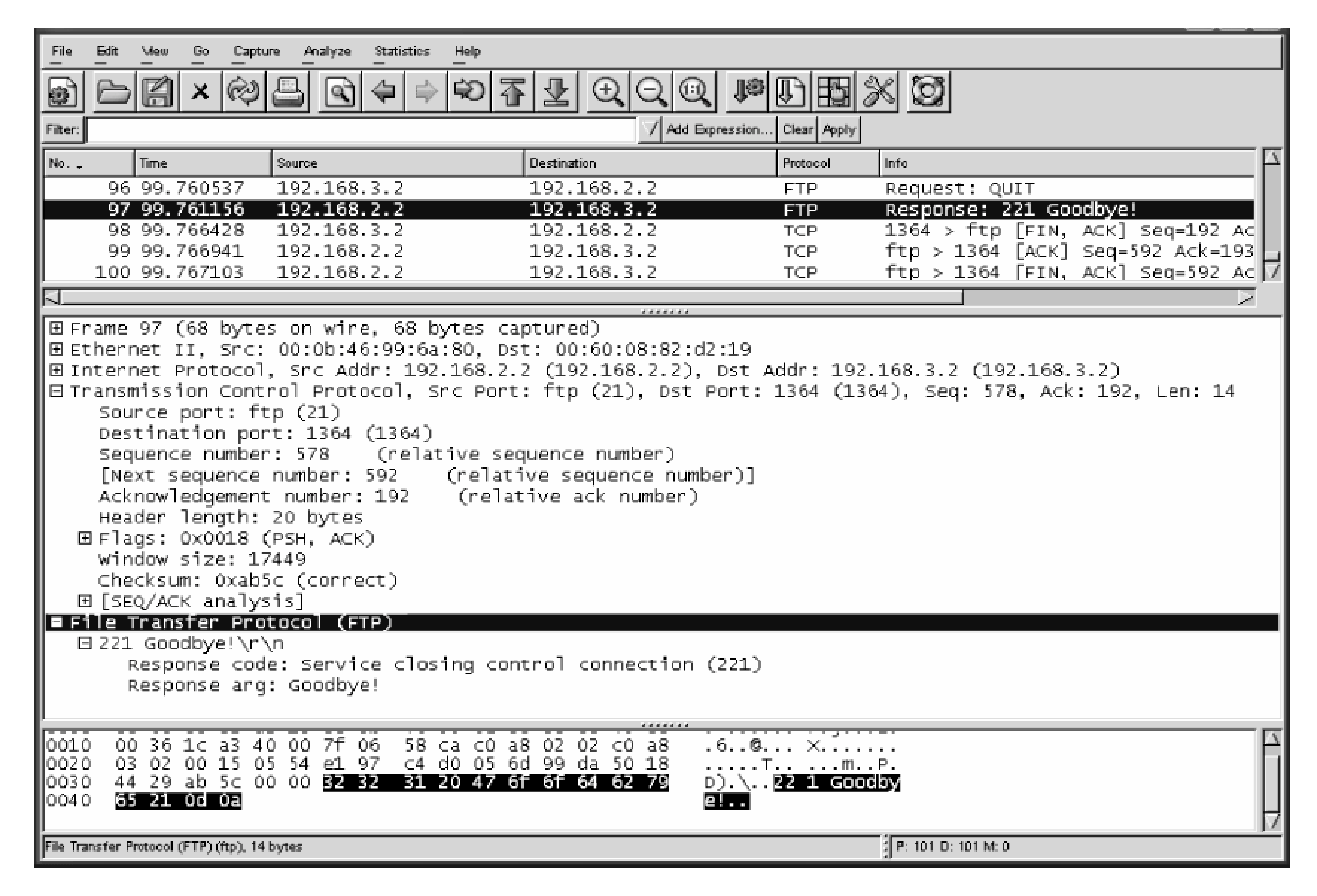



图 8-25

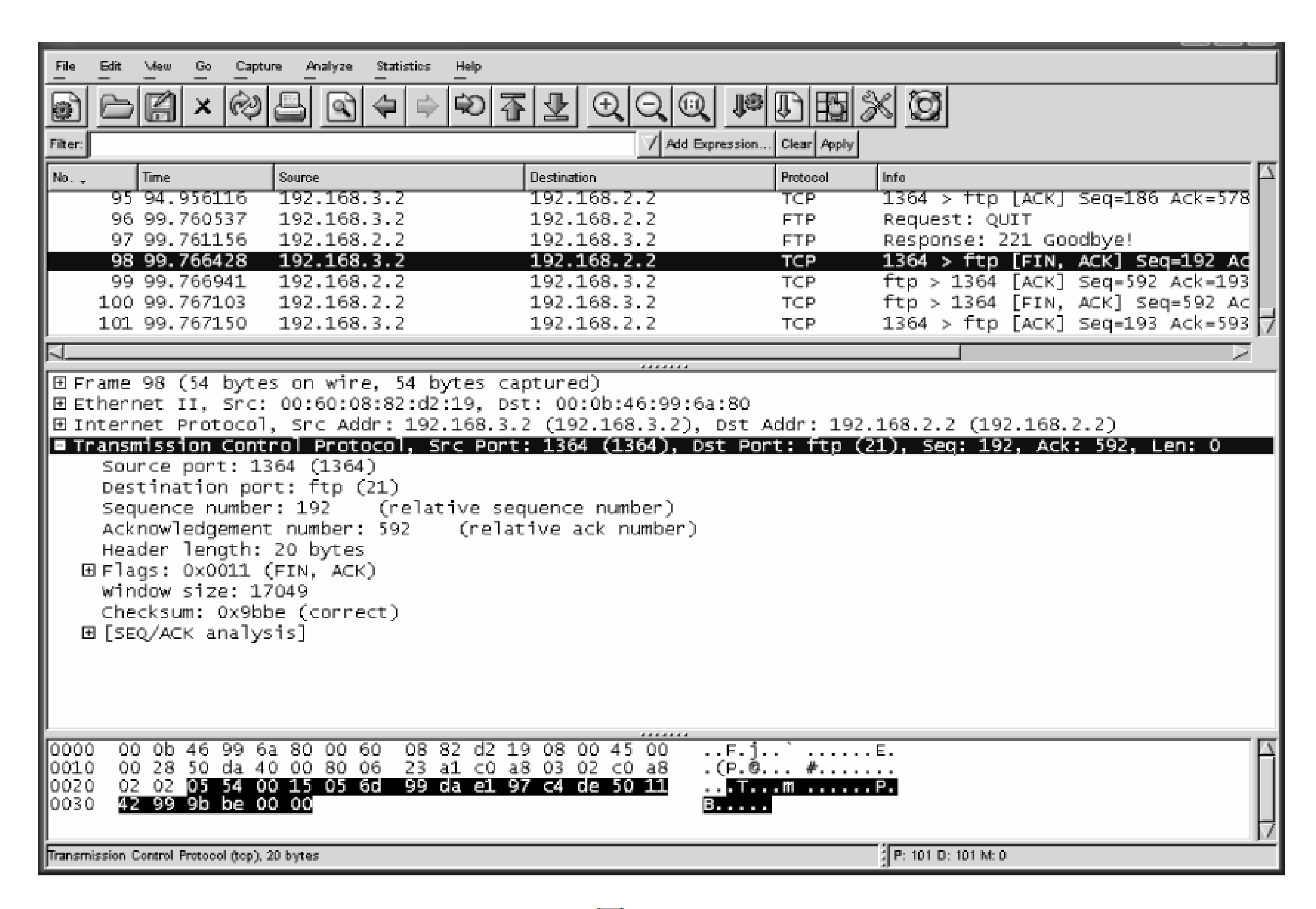



图 8-26

前面讲 FTP 还有个被动模式，它和主动模式的区别如下：

（1）主动模式时，客户端发送一个 PORT 命令给服务器端，服务器依据 PORT 命令中指定的端口和客户端建立数据连接，在被动模式下，客户端发送 PASV(passive)命令给服务器端，表明客户端希望进入 Passive 模式。

（2）服务器端进行应答，应答包括服务器的 IP 地址和一个用于建立数据连接的端口，而不是前面所提到的服务器端用的 20 端口。

（3）客户端发送一个 SYN 包，源端口为客户端自己选择的一个端口，目的端口为服务

器在 PASV 应答命令中指定的端口号。

(4) 服务器端发送 SYN ACK 包给客户端，目的端口为第三步中客户端自己选择的端口，源端口为 PASV 应答中指定的端口号。

(5) 客户端发送一个 ACK 包。可以看到服务器在被动方式下，建立数据连接传输数据时用的端口是由服务器在发送 PASV 应答时指定的，不是主动方式时的 20 端口。其他方面主动和被动方式都一样。

## 8.2 TFTP 协议

TFTP 是普通文件传输协议，它也用于传输文件，但它与 FTP 不同，FTP 使用 TCP 而 TFTP 使用 UDP，默认端口为 69。TFTP 不需要任何形式的用户登录认证。TFTP 设计是为了进行小文件传输，它不具备 FTP 的许多功能，它只能从文件服务器上获得或写入文件，不能列出目录，不能创建和删除目录，不能删除文件，当要终止连接时，TFTP 就会传送小于 512 字节的数据块。如果发送的数据正好为 512 字节的整数倍，那么发送端必须发送一个具有 0 字节的数据块来表示传输结束。

TFTP 数据传输有两种模式：对于 ASCII 文件，用 netascii，这是 8 位的 ASCII 码形式即 NVT ASCII；另一种是对于二进制文件，用 octet，这是二进制 8 位组数据类型。

(1) TFTP 的任何传输由一个读取或写入文件的请求发起，这个请求也是连接请求。如果服务器同意此请求，则服务器打开连接，数据以定长 512 字节传输。每个数据包包含一段数据，服务器发出下一个数据包以前必须得到客户端对上一个数据包的确认。如果数据包在传输过程中丢失，发出方会在超时后重新传输最后一个未被确认的数据包。通信的双方都是数据的发出者与接收者，一方传输数据接收应答，另一方发出应答接收数据。

(2) TFTP 有 5 种类型的包，这 5 种类型的包对应的操作码和用法如表 8-1 所示。

表 8-1 TFTP 操作码

| 操作码 | 包类型 | 说　明 |
|---|---|---|
| 1 | Read Request (RRQ) | 读文件请求，客户端发送 RRQ 报文，服务器响应 DATA 报文 |
| 2 | Write Request (WRQ) | 存储文件请求，服务器响应块号为 0 的 ACK 报文，客户端收到确认后，发送块号为 1 的第一个数据块 |
| 3 | Data (DATA) | 发送数据包文，数据用 DATA 报文发送后，等待 ACK 报文，如果发送端在超时前收到 ACK 报文就发送下一个数据块，否则重传未被确认的数据包文 |
| 4 | Acknowledgment (ACK) | 数据确认报文 |
| 5 | Error (ERROR) | 出错报文 |

TFTP 的会话实例如下。

在主机上向 TFTP 服务器 192.168.2.2 请求一个 ipv6.txt 的文档，如图 8-27 所示。

首先客户端建立连接，使用 UDP 连接 TFTP 服务器的 69 端口，发送一个请求信息，如图 8-28 所示。

在 TFTP 信息中，可以看到，这个请求包包含的字段信息如下。

操作码：1，说明这是个读文件请求 Read Request 报文。

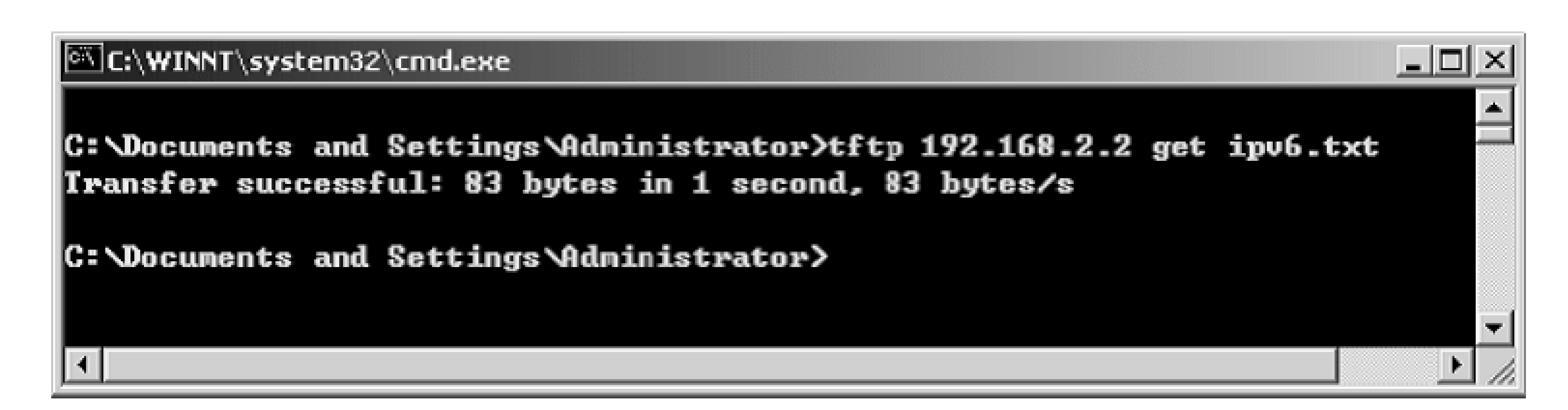



图　8-27

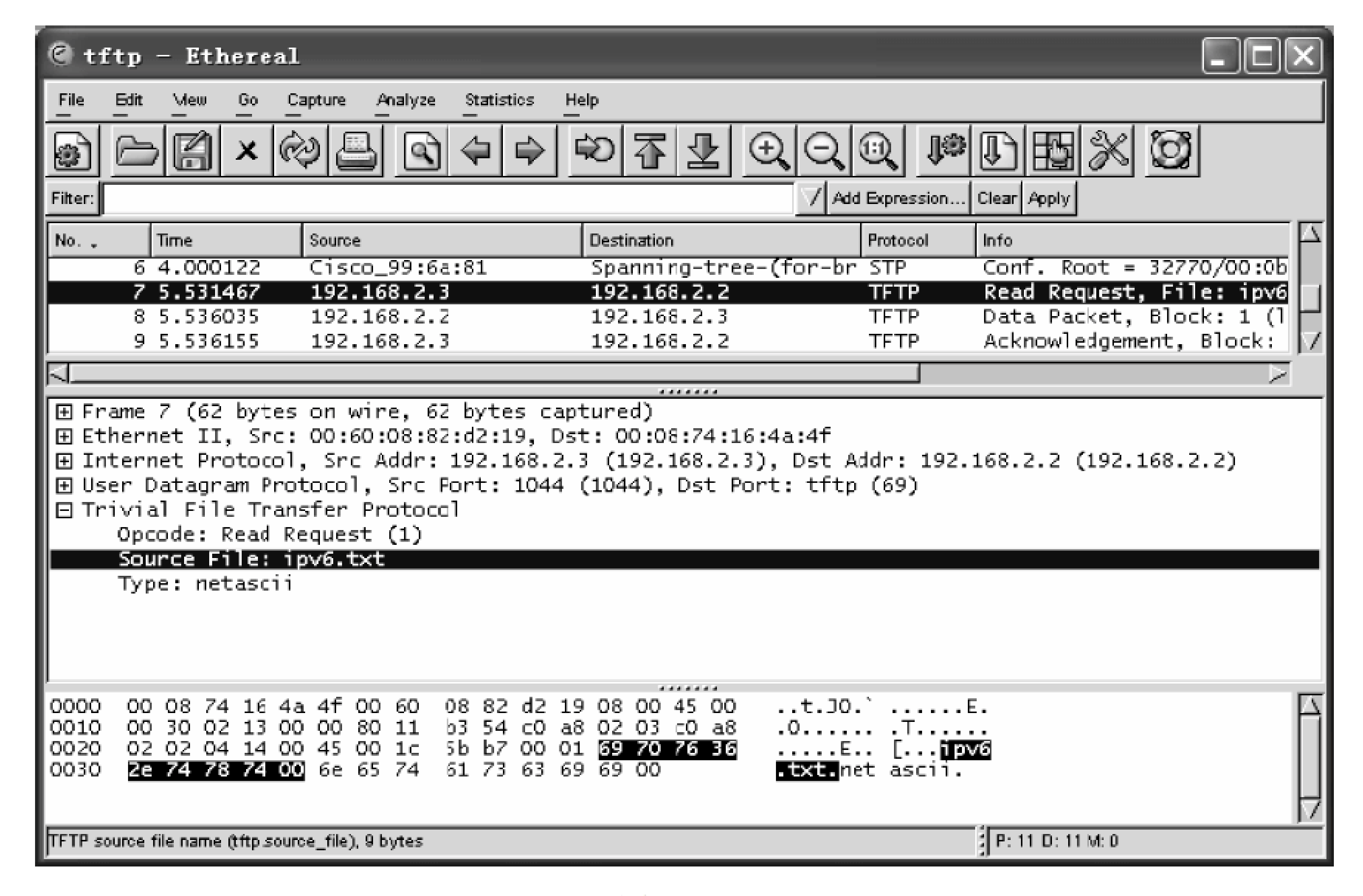



图　8-28

文件名：ipv6.txt，由于文件名长度可变，所以在文件名结束时用 1 字节 0 来作标记。

传输模式：netascii。

随后，如图 8-29 所示，可以看到服务器不是用端口 69 来发送一个数据包，服务器选择了一个端口 2204 来向客户端建立连接时的端口 1044 发送数据包，这时的这个服务器和客户端的端口会一直使用，直到 TFTP 连接结束，数据包包含如下三个字段信息。

操作码：3，说明这是个发送数据包文。

块号：1。

数据：83 bytes。

**注意**：数据超过 512 字节的话，必须分块，数据块号域从 1 开始编码，每个数据块加 1，这样接收方可以确定这个包是新数据还是已经接收过的数据。数据域为 0 字节～512 字节。如果数据域是 512 字节则它不是最后一个包，如果小于 512 字节则表示这个包是最后一个包。除了 ACK 和用于中断的包外，其他的包均要得到确认。发出新的数据包等于确认上次的包。WRQ 和 DATA 包由 ACK 或 ERROR 数据包确认，而 RRQ 数据包由 DATA 或 ERROR 数据包确认。

客户端收到数据后发送一个确认信息，确认信息包含以下字段信息，如图 8-30 所示。

操作码：4，说明这是个数据确认报文 Acknowledgement。

要确认的块号：1。

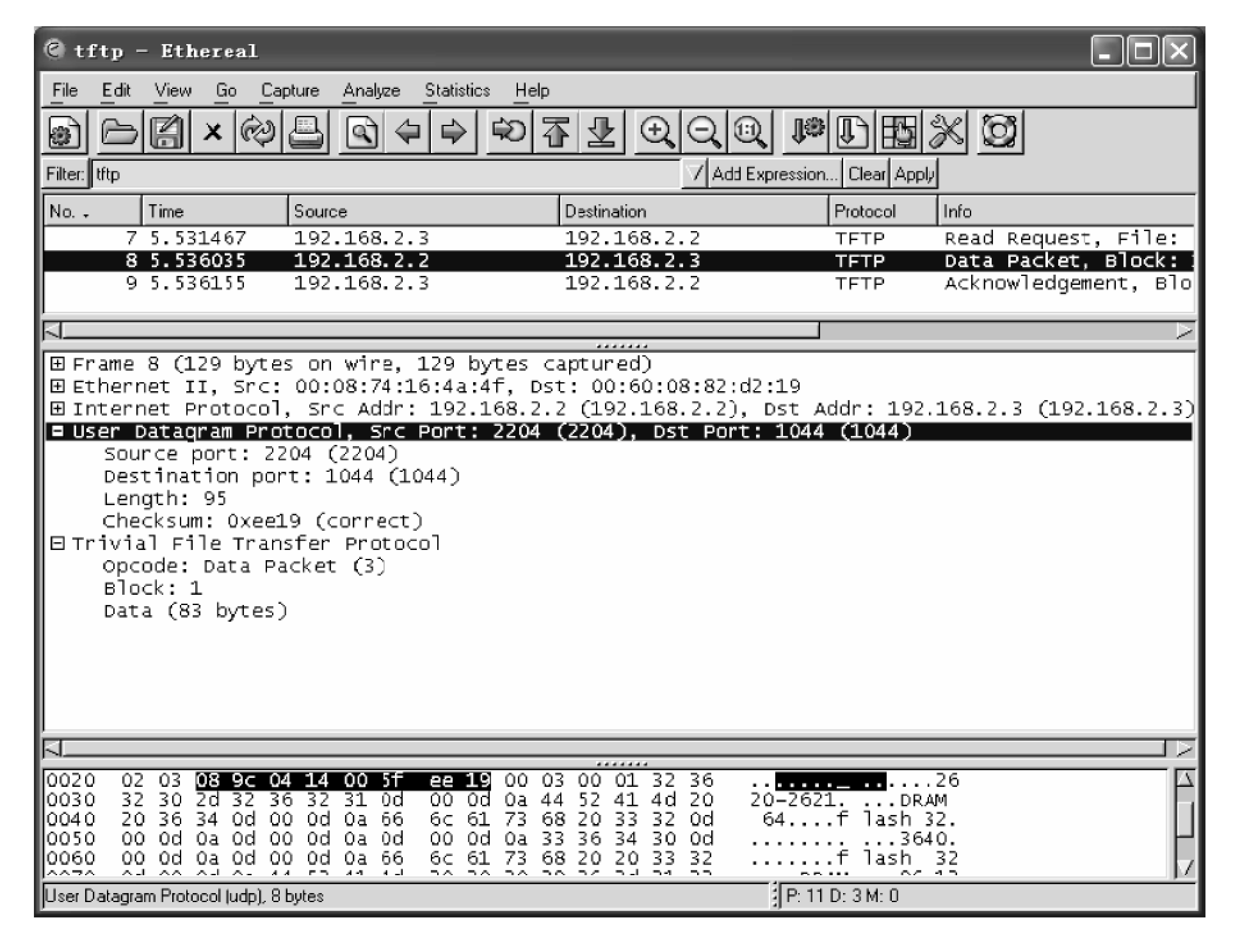



图 8-29

这样就完成了一段数据的传输。

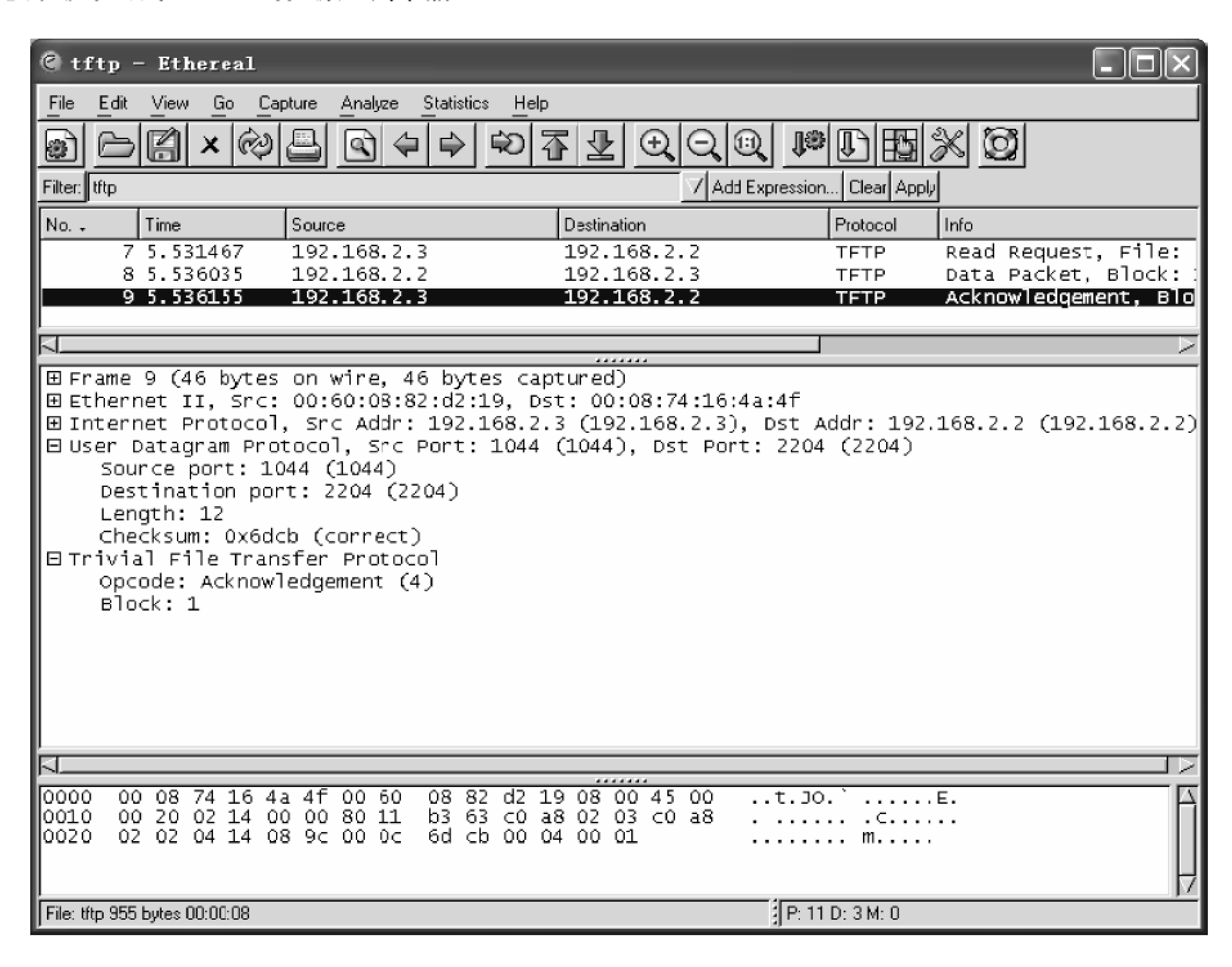



图 8-30

CHAPTER 9

# 第9章 POP3和SMTP

## 9.1 POP3协议

收电子邮件时用到的就是POP3协议。POP3(Post Office Protocol 3)是规定怎样连接到Internet的邮件服务器和下载电子邮件的协议,目前已发展到第3版,称POP3。POP3接收邮件服务器用来接收通过网络发送来的电子邮件,同时POP3允许用户从服务器上把邮件存储到本地主机上,并删除保存在邮件服务器上的邮件。

初始时,服务器通过侦听TCP端口110开始POP3服务。当客户主机需要使用服务时,它将与服务器主机建立TCP连接。当连接建立后,POP3发送确认消息。客户向POP3服务器发送命令并等待响应,命令用ASCII码表示,服务器对客户请求进行响应,响应的第一行以ASCII文本+OK或-ERR指出相应的操作状态是成功还是失败。这个请求/响应过程一直要持续到连接终止。

POP3会话在生命周期中有三个不同的状态。一旦TCP连接被打开,就进入了“确认”状态,大多数现有的POP3客户端与服务器执行采用ASCII明文发送用户名和口令给服务器进行身份确认,客户端必须向POP3服务器确认自己是其客户,服务器确认后,服务器会在邮件上加上排它锁,以防止在进入“更新”状态前对邮件的改变,服务器返回一个“确认”状态码,确认成功后就进入“操作”状态,在“操作”状态下,服务器就获取与客户邮件相关的资源,客户提出服务,完成相应的操作后,客户发出QUIT命令,希望结束会话,如果这时的服务器端处于“操作”状态,则在QUIT命令后进入“更新”状态,删除那些标记成删除的邮件。如果服务器端处于“确认”状态,则结束会话时服务器端不进入“更新”状态。

POP3用12个命令来使得客户端计算机向远程服务器发送执行指令,而服务器则返回给客户端计算机状态码,状态码分别是“确定”(+OK)和“失败”(-ERR)。POP3所有命令以及服务器的响应的状态均以一个<CR><LF>对结束。POP3所有的命令及用法如表9-1所示。

**表 9-1　POP3 的命令及用法**

| 命令 | 参　　数 | 状态 | 描　　述 |
|---|---|---|---|
| USER | Username | 确认 | 用户身份确认是提供用户名 |
| PASS | Password | 确认 | 用户身份确认是提供密码 |
| APOP | Name,Digest | 确认 | 指定邮箱的字串和 MD5 摘要串 |
| STAT | None | 操作 | 请求服务器发回关于邮箱的统计资料,如邮件总数和总字节数 |
| UIDL | Msg# | 操作 | 返回邮件的唯一标识符,POP3 会话的每个标识符都将是唯一的 |
| LIST | Msg# | 操作 | 返回邮件数量和每个邮件的大小 |
| RETR | Msg# | 操作 | 返回由参数标识的邮件的全部文本 |
| DELE | Msg# | 操作 | 服务器将由参数标识的邮件标记为删除,由 Quit 命令执行 |
| RSET | None | 操作 | 服务器将重置所有标记为删除的邮件,用于撤销 DELE 命令 |
| TOP | Msg# | 操作 | 服务器将返回由参数标识的邮件前 n 行内容,n 必须是正整数 |
| NOOP | None | 操作 | 服务器返回一个肯定的响应 |
| QUIT | None | 更新 | 结束会话 |

POP3 的会话实例如下。

开始时,客户端与服务器建立 TCP 连接,如图 9-1～图 9-3 所示,这些 TCP 连接中,服务器端口为 110。

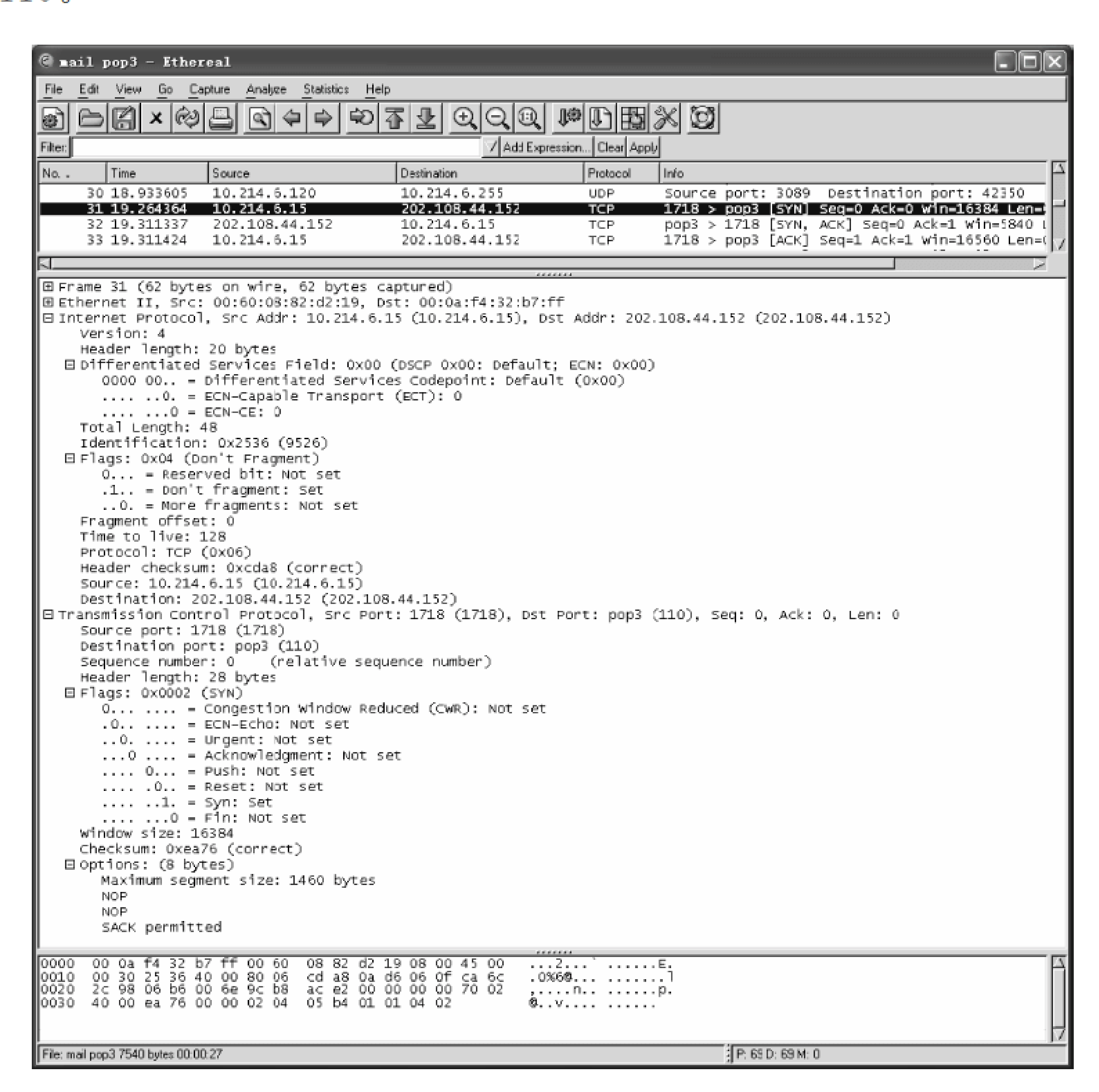



图　9-1

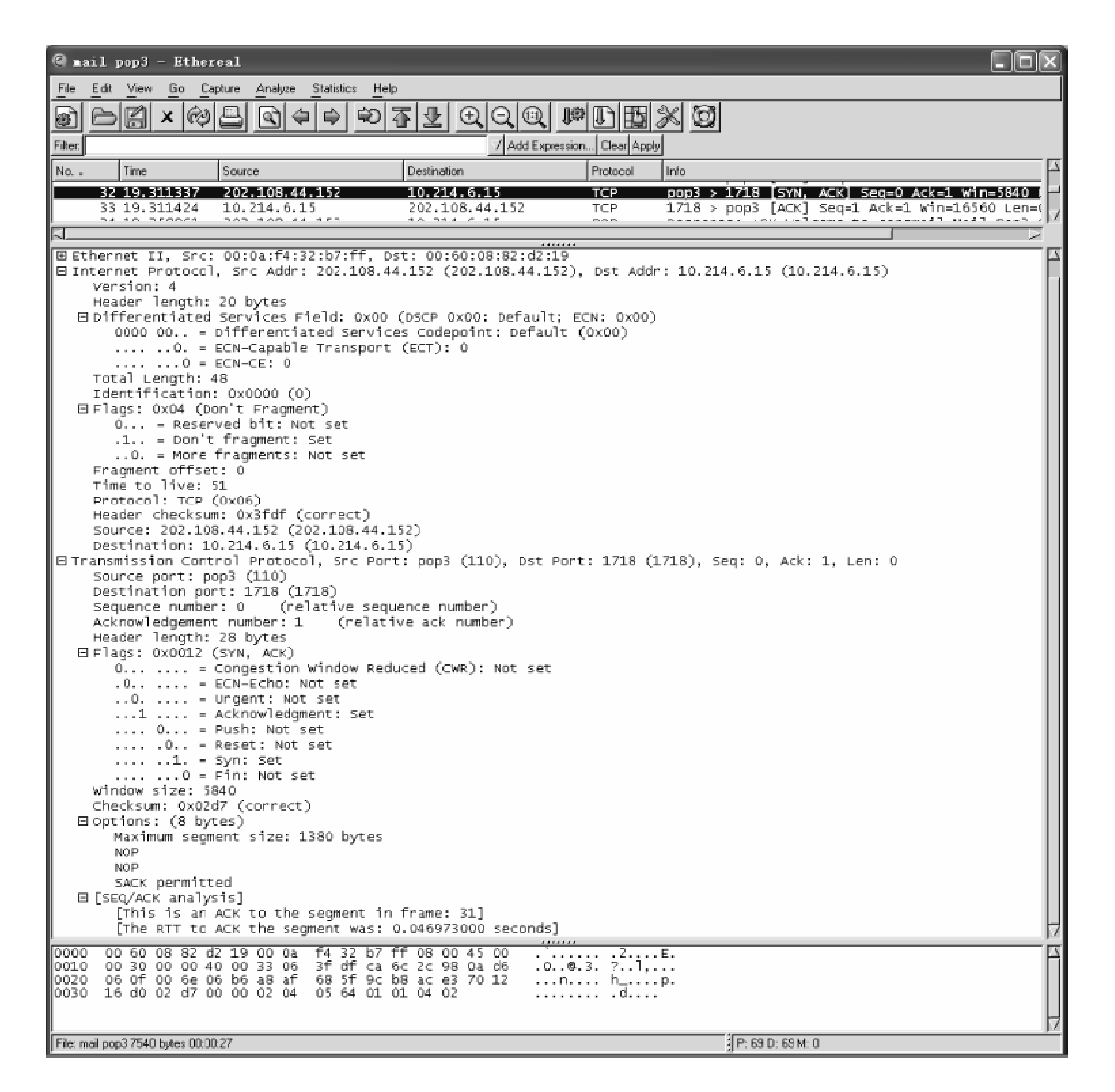

mail pop3 - Ethereal
File Edit View Go Capture Analyze Statistics Help
Filter: Add Expression... Clear Apply
No. Time Source Destination Protocol Info
32 19.311337 202.108.44.152 10.214.6.15 TCP pop3 > 1718 [SYN, ACK] Seq=0 Ack=1 Win=5840
33 19.311424 10.214.6.15 202.108.44.152 TCP 1718 > pop3 [ACK] Seq=1 Ack=1 Win=16560 Len=
Ethernet II, Src: 00:0a:f4:32:b7:ff, Dst: 00:60:08:82:d2:19
Internet Protocol, Src Addr: 202.108.44.152 (202.108.44.152), Dst Addr: 10.214.6.15 (10.214.6.15)
Version: 4
Header length: 20 bytes
Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
0000 00.. = Differentiated Services Codepoint: Default (0x00)
.... ..0. = ECN-Capable Transport (ECT): 0
.... ...0 = ECN-CE: 0
Total Length: 48
Identification: 0x0000 (0)
Flags: 0x04 (Don't Fragment)
0... = Reserved bit: Not set
.1.. = Don't fragment: Set
..0. = More fragments: Not set
Fragment offset: 0
Time to live: 51
Protocol: TCP (0x06)
Header checksum: 0x3fdf (correct)
Source: 202.108.44.152 (202.108.44.152)
Destination: 10.214.6.15 (10.214.6.15)
Transmission Control Protocol, Src Port: pop3 (110), Dst Port: 1718 (1718), Seq: 0, Ack: 1, Len: 0
Source port: pop3 (110)
Destination port: 1718 (1718)
Sequence number: 0 (relative sequence number)
Acknowledgement number: 1 (relative ack number)
Header length: 28 bytes
Flags: 0x0012 (SYN, ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 0... = Push: Not set
.... .0.. = Reset: Not set
.... ..1. = Syn: Set
.... ...0 = Fin: Not set
Window size: 5840
Checksum: 0x02d7 (correct)
Options: (8 bytes)
Maximum segment size: 1380 bytes
NOP
NOP
SACK permitted
[SEQ/ACK analysis]
[This is an ACK to the segment in frame: 31]
[The RTT to ACK the segment was: 0.046973000 seconds]
0000 00 60 08 82 d2 19 00 0a f4 32 b7 ff 08 00 45 00
0010 00 30 00 00 40 00 33 06 3f df ca 6c 2c 98 0a d6
0020 06 0f 00 6e 06 b6 a8 af 68 5f 9c b8 ac e3 70 12
0030 16 d0 02 d7 00 00 02 04 05 64 01 01 04 02
File: mail pop3 7540 bytes 00:00:27
P: 69 D: 69 M: 0

图　9-2

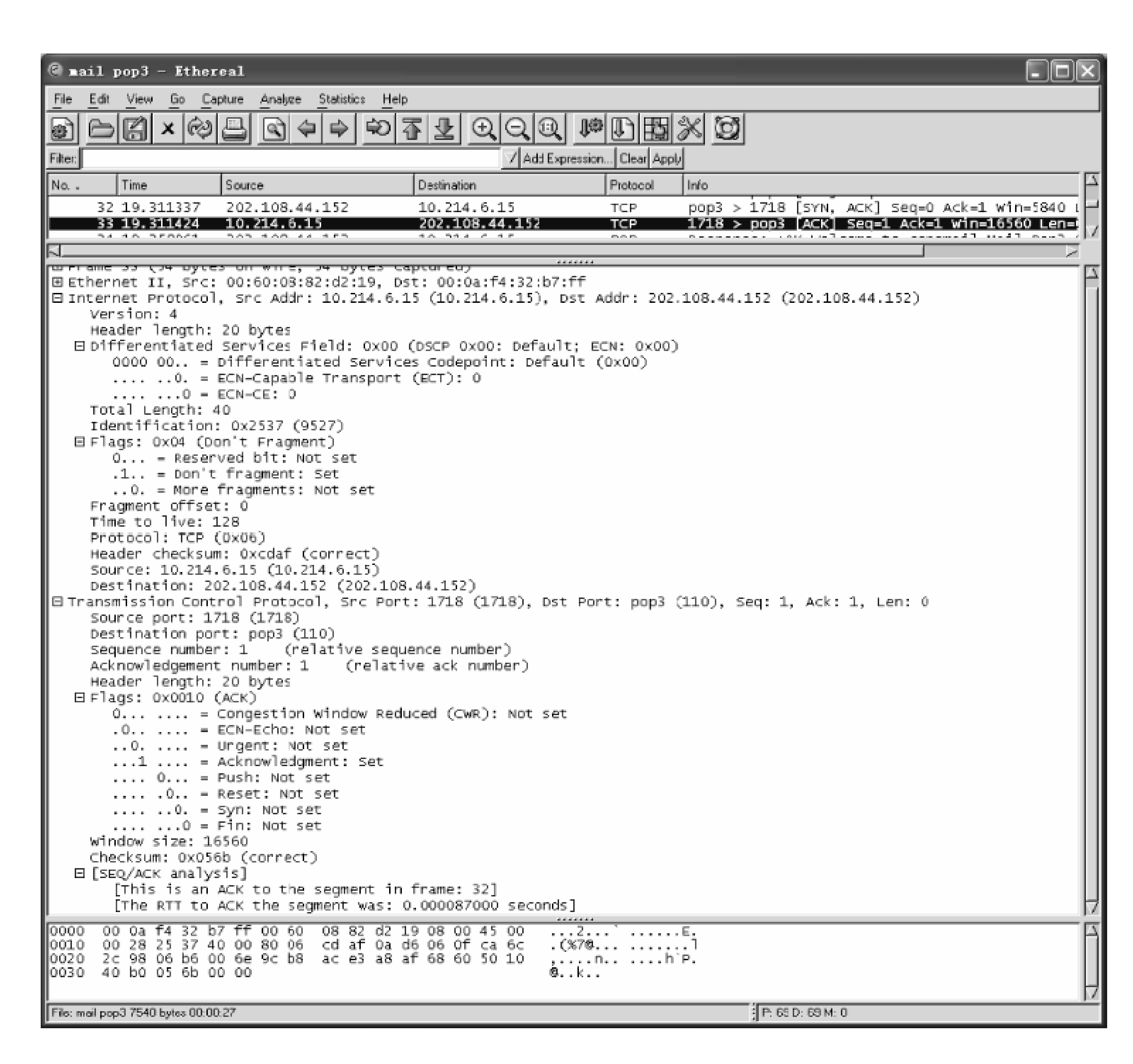

mail pop3 - Ethereal
File Edit View Go Capture Analyze Statistics Help
Filter: Add Expression... Clear Apply
No. Time Source Destination Protocol Info
32 19.311337 202.108.44.152 10.214.6.15 TCP pop3 > 1718 [SYN, ACK] Seq=0 Ack=1 Win=5840
33 19.311424 10.214.6.15 202.108.44.152 TCP 1718 > pop3 [ACK] Seq=1 Ack=1 Win=16560 Len=
Ethernet II, Src: 00:60:08:82:d2:19, Dst: 00:0a:f4:32:b7:ff
Internet Protocol, Src Addr: 10.214.6.15 (10.214.6.15), Dst Addr: 202.108.44.152 (202.108.44.152)
Version: 4
Header length: 20 bytes
Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
0000 00.. = Differentiated Services Codepoint: Default (0x00)
.... ..0. = ECN-Capable Transport (ECT): 0
.... ...0 = ECN-CE: 0
Total Length: 40
Identification: 0x2537 (9527)
Flags: 0x04 (Don't Fragment)
0... = Reserved bit: Not set
.1.. = Don't fragment: Set
..0. = More fragments: Not set
Fragment offset: 0
Time to live: 128
Protocol: TCP (0x06)
Header checksum: 0xcdaf (correct)
Source: 10.214.6.15 (10.214.6.15)
Destination: 202.108.44.152 (202.108.44.152)
Transmission Control Protocol, Src Port: 1718 (1718), Dst Port: pop3 (110), Seq: 1, Ack: 1, Len: 0
Source port: 1718 (1718)
Destination port: pop3 (110)
Sequence number: 1 (relative sequence number)
Acknowledgement number: 1 (relative ack number)
Header length: 20 bytes
Flags: 0x0010 (ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 0... = Push: Not set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 16560
Checksum: 0x056b (correct)
[SEQ/ACK analysis]
[This is an ACK to the segment in frame: 32]
[The RTT to ACK the segment was: 0.000087000 seconds]
0000 00 0a f4 32 b7 ff 00 60 08 82 d2 19 08 00 45 00
0010 00 28 25 37 40 00 80 06 cd af 0a d6 06 0f ca 6c
0020 2c 98 06 b6 00 6e 9c b8 ac e3 a8 af 68 60 50 10
0030 40 b0 05 6b 00 00
File: mail pop3 7540 bytes 00:00:27
P: 69 D: 69 M: 0

图　9-3

经过三次握手后，服务器发送一个 POP3 响应，进入确认状态，可以看到一个 POP3 响应由一个状态码和一些附加信息组成，如图 9-4 所示，状态码为 OK，这个响应信息以<CRLF>结束，在图中用\r\n 来表示。

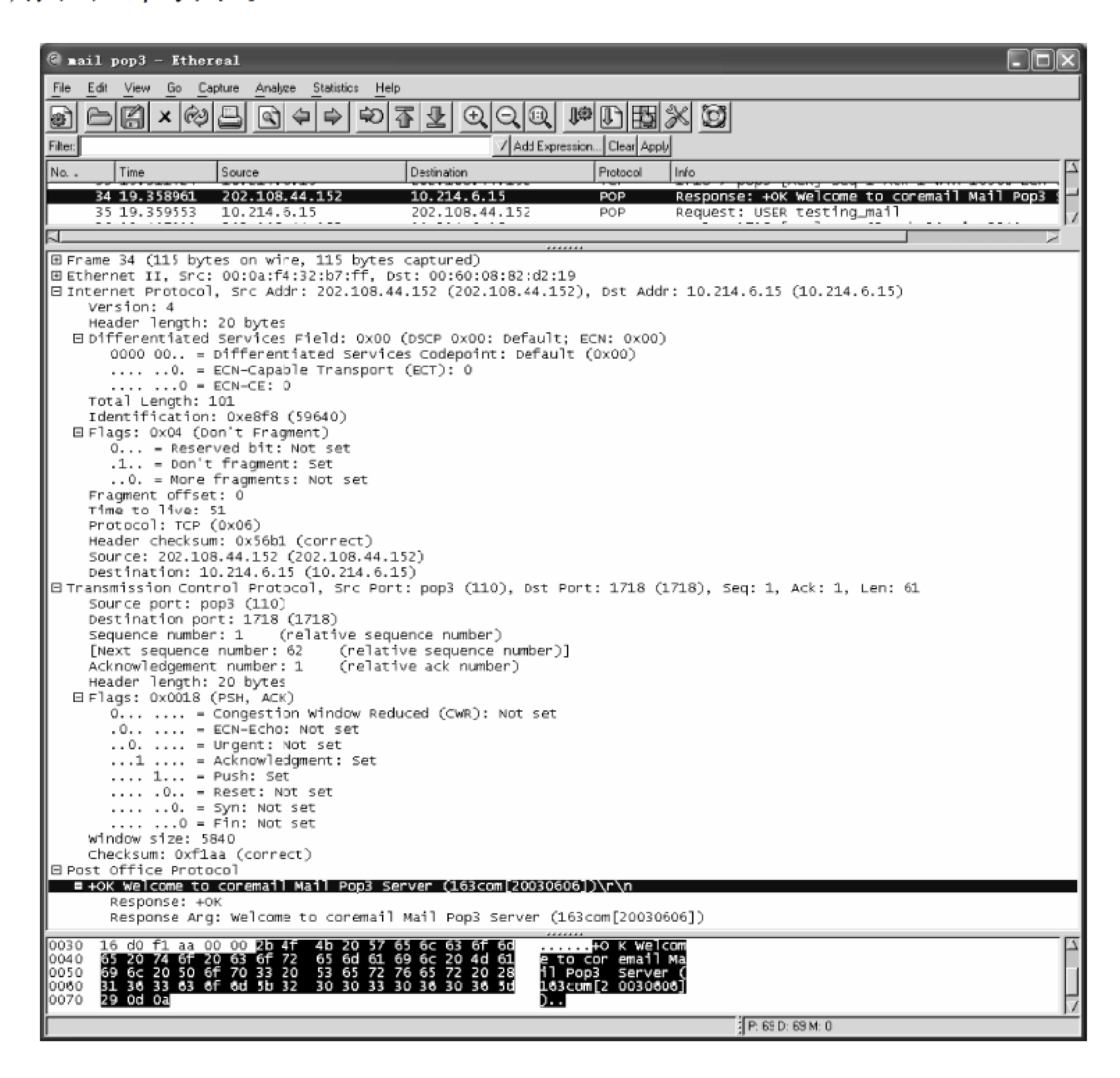



图 9-4

从图 9-5 中可以看到，进入确认状态后，客户必须向服务器证明它的身份，那么客户端就发送一个 USER 命令，输入用户名。

服务器端发送 TCP 确认，如图 9-6 所示。

服务器返回一个状态码 OK，表示服务器接受客户端计算机的 user 命令，如图 9-7 所示。

客户端发送 PASS 命令，输入密码，如图 9-8 所示。

若用户名和密码正确，则服务器返回一个消息，状态码为 OK，同时，还可以看到邮箱里信件的信息：一封邮件，大小为 807B，如图 9-9 所示。

客户端发送 STAT 命令来获得目前邮箱中电子邮件的数量和每个邮件的大小，如图 9-10 所示。

服务器回应 STAT 请求，状态码为 OK，如图 9-11 所示。

客户端发送 LIST 命令来获得邮箱里邮件的编号，如图 9-12 所示，Response Arg 1807 表示有一封信，大小为 807B。

**注意**：LIST 命令的参数可选，该参数是一个数字，表示的是邮件在邮箱中的编号，可以利用不带参数的 LIST 命令，获得各邮件的编号。

服务器回应 LIST 请求，状态码为 OK，如图 9-13 所示。

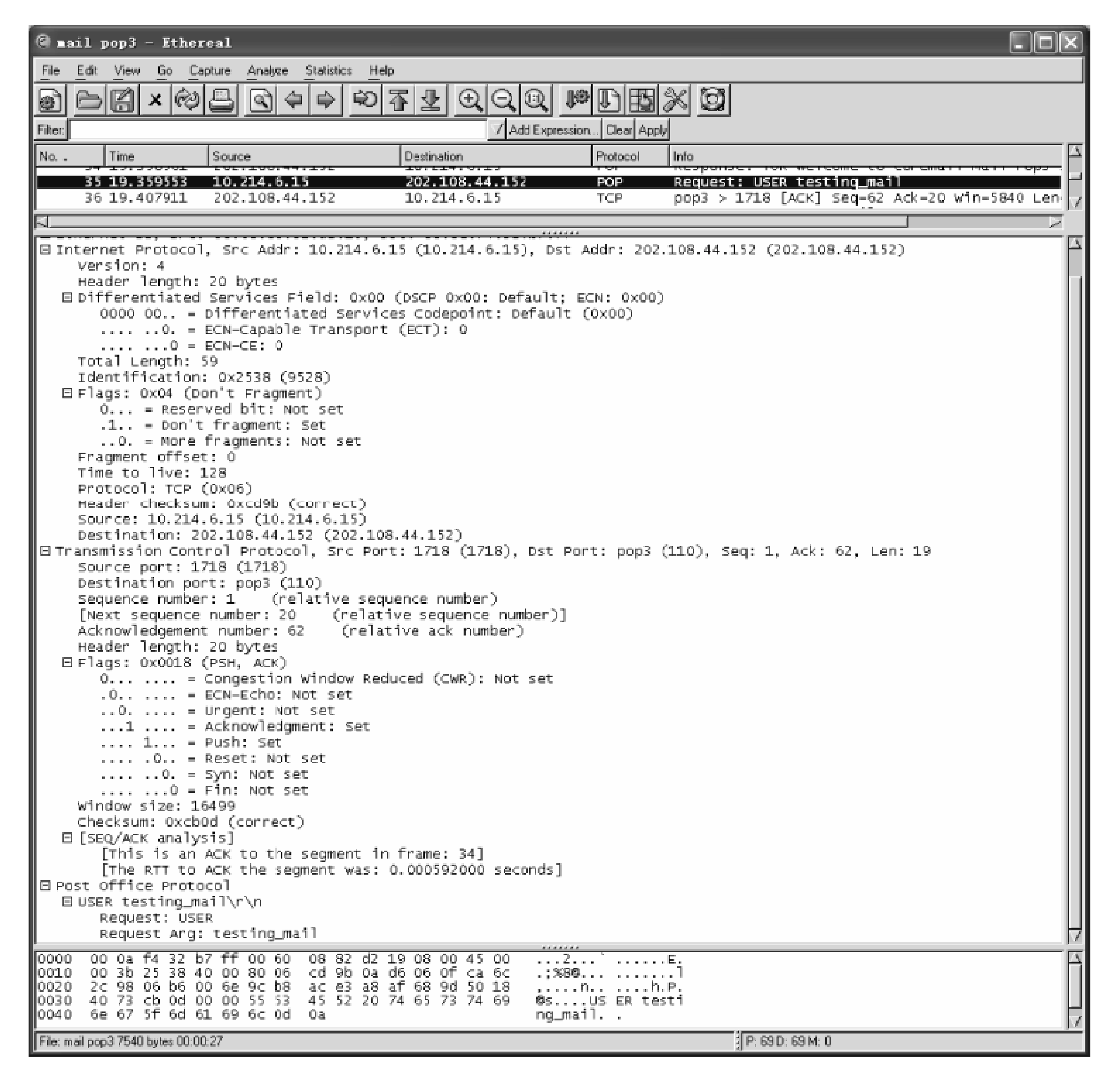

mail pop3 - Ethereal
File Edit View Go Capture Analyze Statistics Help
Filter: Add Expression... Clear Apply
No. Time Source Destination Protocol Info
35 19.359553 10.214.6.15 202.108.44.152 POP Request: USER testing_mail
36 19.407911 202.108.44.152 10.214.6.15 TCP pop3 > 1718 [ACK] Seq=62 Ack=20 Win=5840 Len
Internet Protocol, Src Addr: 10.214.6.15 (10.214.6.15), Dst Addr: 202.108.44.152 (202.108.44.152)
Version: 4
Header length: 20 bytes
Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
0000 00.. = Differentiated Services Codepoint: Default (0x00)
.... ..0. = ECN-Capable Transport (ECT): 0
.... ...0 = ECN-CE: 0
Total Length: 59
Identification: 0x2538 (9528)
Flags: 0x04 (Don't Fragment)
0... = Reserved bit: Not set
.1.. = Don't fragment: Set
..0. = More fragments: Not set
Fragment offset: 0
Time to live: 128
Protocol: TCP (0x06)
Header checksum: 0xcd9b (correct)
Source: 10.214.6.15 (10.214.6.15)
Destination: 202.108.44.152 (202.108.44.152)
Transmission Control Protocol, Src Port: 1718 (1718), Dst Port: pop3 (110), Seq: 1, Ack: 62, Len: 19
Source port: 1718 (1718)
Destination port: pop3 (110)
Sequence number: 1 (relative sequence number)
[Next sequence number: 20 (relative sequence number)]
Acknowledgement number: 62 (relative ack number)
Header length: 20 bytes
Flags: 0x0018 (PSH, ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 1... = Push: Set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 16499
Checksum: 0xcb0d (correct)
[SEQ/ACK analysis]
[This is an ACK to the segment in frame: 34]
[The RTT to ACK the segment was: 0.000592000 seconds]
Post Office Protocol
USER testing_mail\r\n
Request: USER
Request Arg: testing_mail
0000 00 0a f4 32 b7 ff 00 60 08 82 d2 19 08 00 45 00
0010 00 3b 25 38 40 00 80 06 cd 9b 0a d6 06 0f ca 6c
0020 2c 98 06 b6 00 6e 9c b8 ac e3 a8 af 68 9d 50 18
0030 40 73 cb 0d 00 00 55 53 45 52 20 74 65 73 74 69
0040 6e 67 5f 6d 61 69 6c 0d 0a
File: mail pop3 7540 bytes 00:00:27
P: 69 D: 69 M: 0

图　9-5

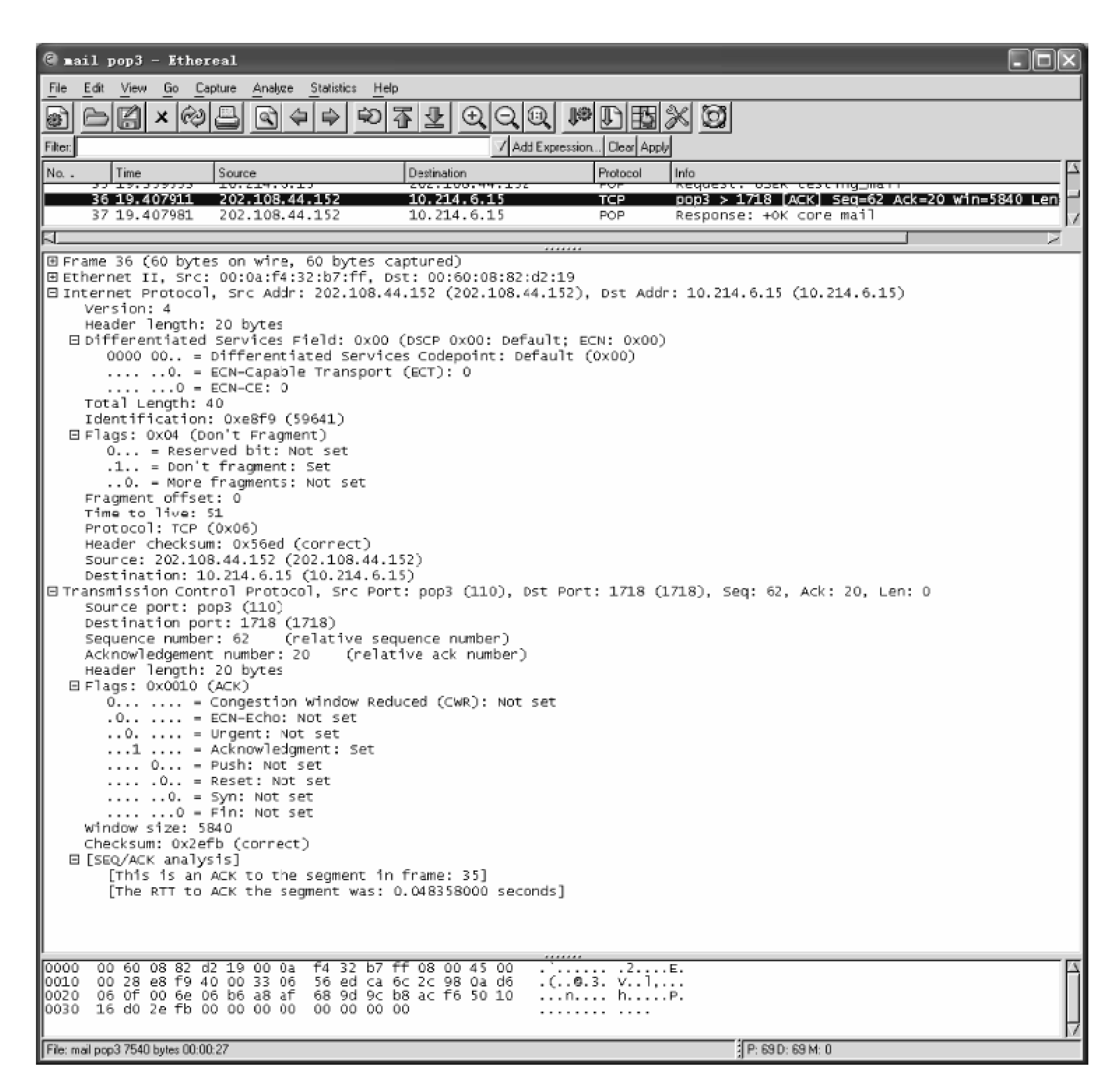

mail pop3 - Ethereal
File Edit View Go Capture Analyze Statistics Help
Filter: Add Expression... Clear Apply
No. Time Source Destination Protocol Info
36 19.407911 202.108.44.152 10.214.6.15 TCP pop3 > 1718 [ACK] Seq=62 Ack=20 Win=5840 Len
37 19.407981 202.108.44.152 10.214.6.15 POP Response: +OK core mail
Frame 36 (60 bytes on wire, 60 bytes captured)
Ethernet II, Src: 00:0a:f4:32:b7:ff, Dst: 00:60:08:82:d2:19
Internet Protocol, Src Addr: 202.108.44.152 (202.108.44.152), Dst Addr: 10.214.6.15 (10.214.6.15)
Version: 4
Header length: 20 bytes
Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
0000 00.. = Differentiated Services Codepoint: Default (0x00)
.... ..0. = ECN-Capable Transport (ECT): 0
.... ...0 = ECN-CE: 0
Total Length: 40
Identification: 0xe8f9 (59641)
Flags: 0x04 (Don't Fragment)
0... = Reserved bit: Not set
.1.. = Don't fragment: Set
..0. = More fragments: Not set
Fragment offset: 0
Time to live: 51
Protocol: TCP (0x06)
Header checksum: 0x56ed (correct)
Source: 202.108.44.152 (202.108.44.152)
Destination: 10.214.6.15 (10.214.6.15)
Transmission Control Protocol, Src Port: pop3 (110), Dst Port: 1718 (1718), Seq: 62, Ack: 20, Len: 0
Source port: pop3 (110)
Destination port: 1718 (1718)
Sequence number: 62 (relative sequence number)
Acknowledgement number: 20 (relative ack number)
Header length: 20 bytes
Flags: 0x0010 (ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 0... = Push: Not set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 5840
Checksum: 0x2efb (correct)
[SEQ/ACK analysis]
[This is an ACK to the segment in frame: 35]
[The RTT to ACK the segment was: 0.048358000 seconds]
0000 00 60 08 82 d2 19 00 0a f4 32 b7 ff 08 00 45 00
0010 00 28 e8 f9 40 00 33 06 56 ed ca 6c 2c 98 0a d6
0020 06 0f 00 6e 06 b6 a8 af 68 9d 9c b8 ac f6 50 10
0030 16 d0 2e fb 00 00 00 00 00 00 00 00
File: mail pop3 7540 bytes 00:00:27
P: 69 D: 69 M: 0

图　9-6

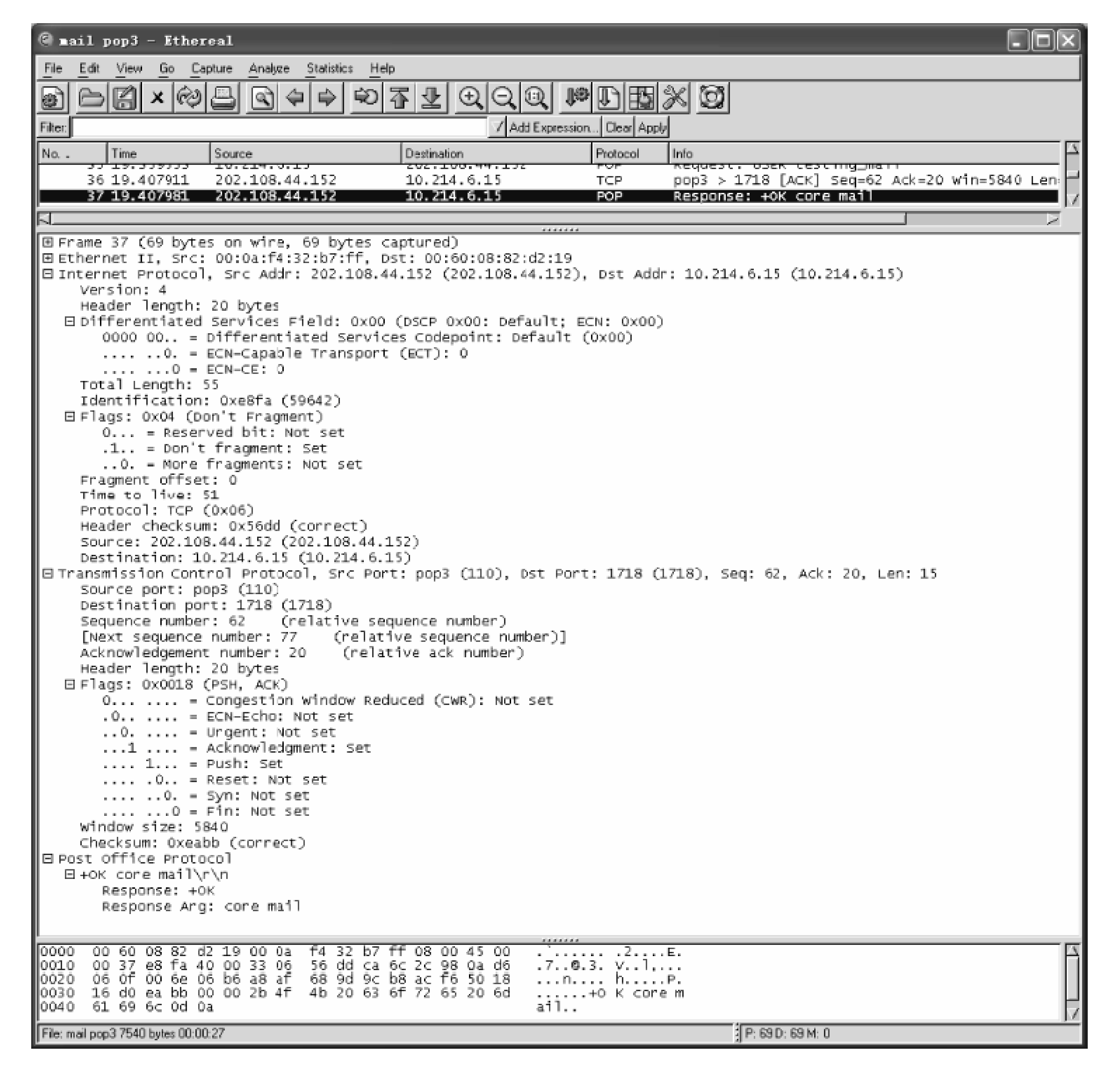

mail pop3 - Ethereal
File Edit View Go Capture Analyze Statistics Help
Filter: Add Expression... Clear Apply
No. . Time Source Destination Protocol Info
36 19.407911 202.108.44.152 10.214.6.15 TCP pop3 > 1718 [ACK] Seq=62 Ack=20 Win=5840 Len
37 19.407981 202.108.44.152 10.214.6.15 POP Response: +OK core mail
Frame 37 (69 bytes on wire, 69 bytes captured)
Ethernet II, Src: 00:0a:f4:32:b7:ff, Dst: 00:60:08:82:d2:19
Internet Protocol, Src Addr: 202.108.44.152 (202.108.44.152), Dst Addr: 10.214.6.15 (10.214.6.15)
Version: 4
Header length: 20 bytes
Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
0000 00.. = Differentiated Services Codepoint: Default (0x00)
.... ..0. = ECN-Capable Transport (ECT): 0
.... ...0 = ECN-CE: 0
Total Length: 55
Identification: 0xe8fa (59642)
Flags: 0x04 (Don't Fragment)
0... = Reserved bit: Not set
.1.. = Don't fragment: Set
..0. = More fragments: Not set
Fragment offset: 0
Time to live: 51
Protocol: TCP (0x06)
Header checksum: 0x56dd (correct)
Source: 202.108.44.152 (202.108.44.152)
Destination: 10.214.6.15 (10.214.6.15)
Transmission Control Protocol, Src Port: pop3 (110), Dst Port: 1718 (1718), Seq: 62, Ack: 20, Len: 15
Source port: pop3 (110)
Destination port: 1718 (1718)
Sequence number: 62 (relative sequence number)
[Next sequence number: 77 (relative sequence number)]
Acknowledgement number: 20 (relative ack number)
Header length: 20 bytes
Flags: 0x0018 (PSH, ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 1... = Push: Set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 5840
Checksum: 0xeabb (correct)
Post Office Protocol
+OK core mail\r\n
Response: +OK
Response Arg: core mail
0000 00 60 08 82 d2 19 00 0a f4 32 b7 ff 08 00 45 00
0010 00 37 e8 fa 40 00 33 06 56 dd ca 6c 2c 98 0a d6
0020 06 0f 00 6e 06 b6 a8 af 68 9d 9c b8 ac f6 50 18
0030 16 d0 ea bb 00 00 2b 4f 4b 20 63 6f 72 65 20 6d ......+O K core m
0040 61 69 6c 0d 0a ail..
File: mail pop3 7540 bytes 00:00:27
P: 69 D: 69 M: 0

图 9-7


mail pop3 - Ethereal
File Edit View Go Capture Analyze Statistics Help
Filter: Add Expression... Clear Apply
No. . Time Source Destination Protocol Info
38 19.408286 10.214.6.15 202.108.44.152 POP Request: PASS testing
39 19.462318 202.108.44.152 10.214.6.15 POP Response: +OK 1 message(s) [807 byte(s)]
Internet Protocol, Src Addr: 10.214.6.15 (10.214.6.15), Dst Addr: 202.108.44.152 (202.108.44.152)
Version: 4
Header length: 20 bytes
Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
0000 00.. = Differentiated Services Codepoint: Default (0x00)
.... ..0. = ECN-Capable Transport (ECT): 0
.... ...0 = ECN-CE: 0
Total Length: 54
Identification: 0x2539 (9529)
Flags: 0x04 (Don't Fragment)
0... = Reserved bit: Not set
.1.. = Don't fragment: Set
..0. = More fragments: Not set
Fragment offset: 0
Time to live: 128
Protocol: TCP (0x06)
Header checksum: 0xcd9f (correct)
Source: 10.214.6.15 (10.214.6.15)
Destination: 202.108.44.152 (202.108.44.152)
Transmission Control Protocol, Src Port: 1718 (1718), Dst Port: pop3 (110), Seq: 20, Ack: 77, Len: 14
Source port: 1718 (1718)
Destination port: pop3 (110)
Sequence number: 20 (relative sequence number)
[Next sequence number: 34 (relative sequence number)]
Acknowledgement number: 77 (relative ack number)
Header length: 20 bytes
Flags: 0x0018 (PSH, ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 1... = Push: Set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 16484
Checksum: 0xebea (correct)
[SEQ/ACK analysis]
[This is an ACK to the segment in frame: 37]
[The RTT to ACK the segment was: 0.000305000 seconds]
Post Office Protocol
PASS testing\r\n
Request: PASS
Request Arg: testing
0000 00 0a f4 32 b7 ff 00 60 08 82 d2 19 08 00 45 00
0010 00 36 25 39 40 00 80 06 cd 9f 0a d6 06 0f ca 6c
0020 2c 98 06 b6 00 6e 9c b8 ac f6 a8 af 68 ac 50 18
0030 40 64 eb ea 00 00 50 41 53 53 20 74 65 73 74 69 @d....PA SS testi
0040 6e 67 0d 0a ng..
File: mail pop3 7540 bytes 00:00:27
P: 69 D: 69 M: 0

图 9-8

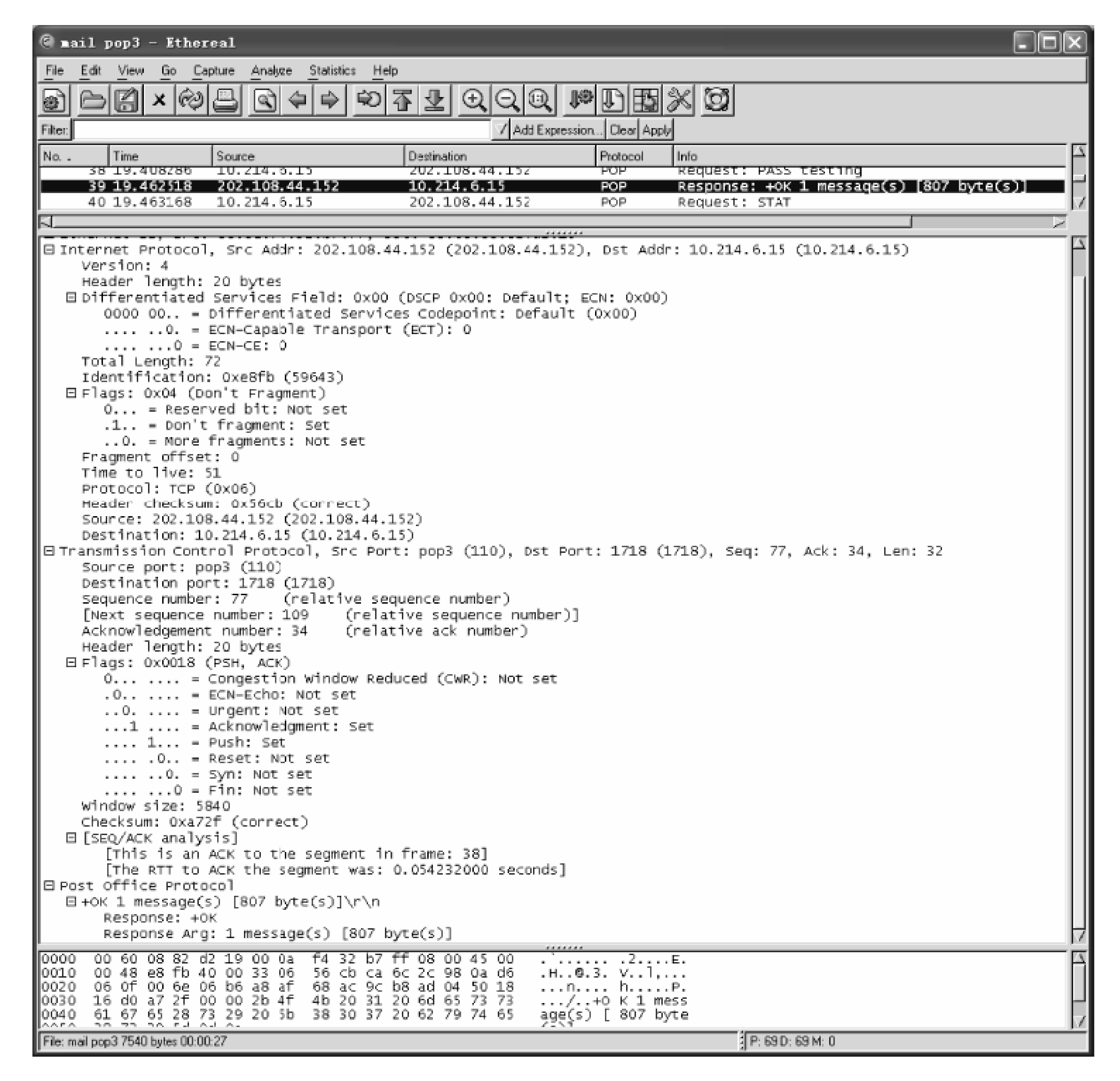

mail pop3 - Ethereal
File Edit View Go Capture Analyze Statistics Help
Filter: Add Expression... Clear Apply
No. . Time Source Destination Protocol Info
38 19.408286 10.214.6.15 202.108.44.152 POP Request: PASS testing
39 19.462518 202.108.44.152 10.214.6.15 POP Response: +OK 1 message(s) [807 byte(s)]
40 19.463168 10.214.6.15 202.108.44.152 POP Request: STAT
Internet Protocol, Src Addr: 202.108.44.152 (202.108.44.152), Dst Addr: 10.214.6.15 (10.214.6.15)
Version: 4
Header length: 20 bytes
Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
0000 00.. = Differentiated Services Codepoint: Default (0x00)
.... ..0. = ECN-Capable Transport (ECT): 0
.... ...0 = ECN-CE: 0
Total Length: 72
Identification: 0xe8fb (59643)
Flags: 0x04 (Don't Fragment)
0... = Reserved bit: Not set
.1.. = Don't fragment: Set
..0. = More fragments: Not set
Fragment offset: 0
Time to live: 51
Protocol: TCP (0x06)
Header checksum: 0x56cb (correct)
Source: 202.108.44.152 (202.108.44.152)
Destination: 10.214.6.15 (10.214.6.15)
Transmission Control Protocol, Src Port: pop3 (110), Dst Port: 1718 (1718), Seq: 77, Ack: 34, Len: 32
Source port: pop3 (110)
Destination port: 1718 (1718)
Sequence number: 77 (relative sequence number)
[Next sequence number: 109 (relative sequence number)]
Acknowledgement number: 34 (relative ack number)
Header length: 20 bytes
Flags: 0x0018 (PSH, ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 1... = Push: Set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 5840
Checksum: 0xa72f (correct)
[SEQ/ACK analysis]
[This is an ACK to the segment in frame: 38]
[The RTT to ACK the segment was: 0.054232000 seconds]
Post Office Protocol
+OK 1 message(s) [807 byte(s)]\r\n
Response: +OK
Response Arg: 1 message(s) [807 byte(s)]
0000 00 60 08 82 d2 19 00 0a f4 32 b7 ff 08 00 45 00 .`.......2....E.
0010 00 48 e8 fb 40 00 33 06 56 cb ca 6c 2c 98 0a d6 .H..@.3. V..l,...
0020 06 0f 00 6e 06 b6 a8 af 68 ac 9c b8 ad 04 50 18 ...n.... h.....P.
0030 16 d0 a7 2f 00 00 2b 4f 4b 20 31 20 6d 65 73 73 .../..+O K 1 mess
0040 61 67 65 28 73 29 20 5b 38 30 37 20 62 79 74 65 age(s) [ 807 byte
File: mail pop3 7540 bytes 00:00:27
P: 69 D: 69 M: 0

图 9-9

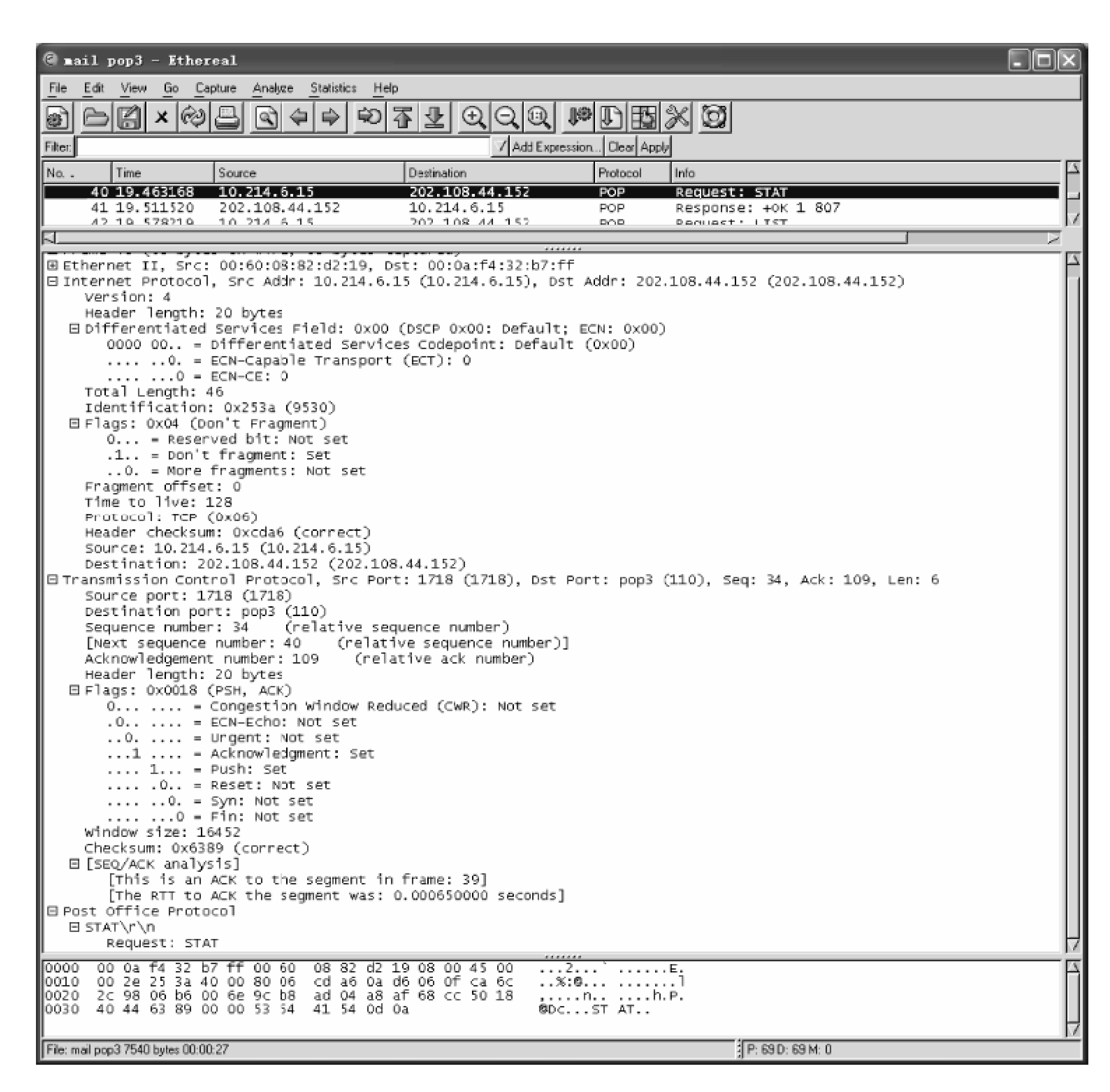

mail pop3 - Ethereal
File Edit View Go Capture Analyze Statistics Help
Filter: Add Expression... Clear Apply
No. . Time Source Destination Protocol Info
40 19.463168 10.214.6.15 202.108.44.152 POP Request: STAT
41 19.511520 202.108.44.152 10.214.6.15 POP Response: +OK 1 807
Ethernet II, Src: 00:60:08:82:d2:19, Dst: 00:0a:f4:32:b7:ff
Internet Protocol, Src Addr: 10.214.6.15 (10.214.6.15), Dst Addr: 202.108.44.152 (202.108.44.152)
Version: 4
Header length: 20 bytes
Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
0000 00.. = Differentiated Services Codepoint: Default (0x00)
.... ..0. = ECN-Capable Transport (ECT): 0
.... ...0 = ECN-CE: 0
Total Length: 46
Identification: 0x253a (9530)
Flags: 0x04 (Don't Fragment)
0... = Reserved bit: Not set
.1.. = Don't fragment: Set
..0. = More fragments: Not set
Fragment offset: 0
Time to live: 128
Protocol: TCP (0x06)
Header checksum: 0xcda6 (correct)
Source: 10.214.6.15 (10.214.6.15)
Destination: 202.108.44.152 (202.108.44.152)
Transmission Control Protocol, Src Port: 1718 (1718), Dst Port: pop3 (110), Seq: 34, Ack: 109, Len: 6
Source port: 1718 (1718)
Destination port: pop3 (110)
Sequence number: 34 (relative sequence number)
[Next sequence number: 40 (relative sequence number)]
Acknowledgement number: 109 (relative ack number)
Header length: 20 bytes
Flags: 0x0018 (PSH, ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 1... = Push: Set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 16452
Checksum: 0x6389 (correct)
[SEQ/ACK analysis]
[This is an ACK to the segment in frame: 39]
[The RTT to ACK the segment was: 0.000650000 seconds]
Post Office Protocol
STAT\r\n
Request: STAT
0000 00 0a f4 32 b7 ff 00 60 08 82 d2 19 08 00 45 00 ...2...` ......E.
0010 00 2e 25 3a 40 00 80 06 cd a6 0a d6 06 0f ca 6c ..%:@... .......l
0020 2c 98 06 b6 00 6e 9c b8 ad 04 a8 af 68 cc 50 18 ,....n.. ....h.P.
0030 40 44 63 89 00 00 53 54 41 54 0d 0a @Dc...ST AT..
File: mail pop3 7540 bytes 00:00:27
P: 69 D: 69 M: 0

图 9-10

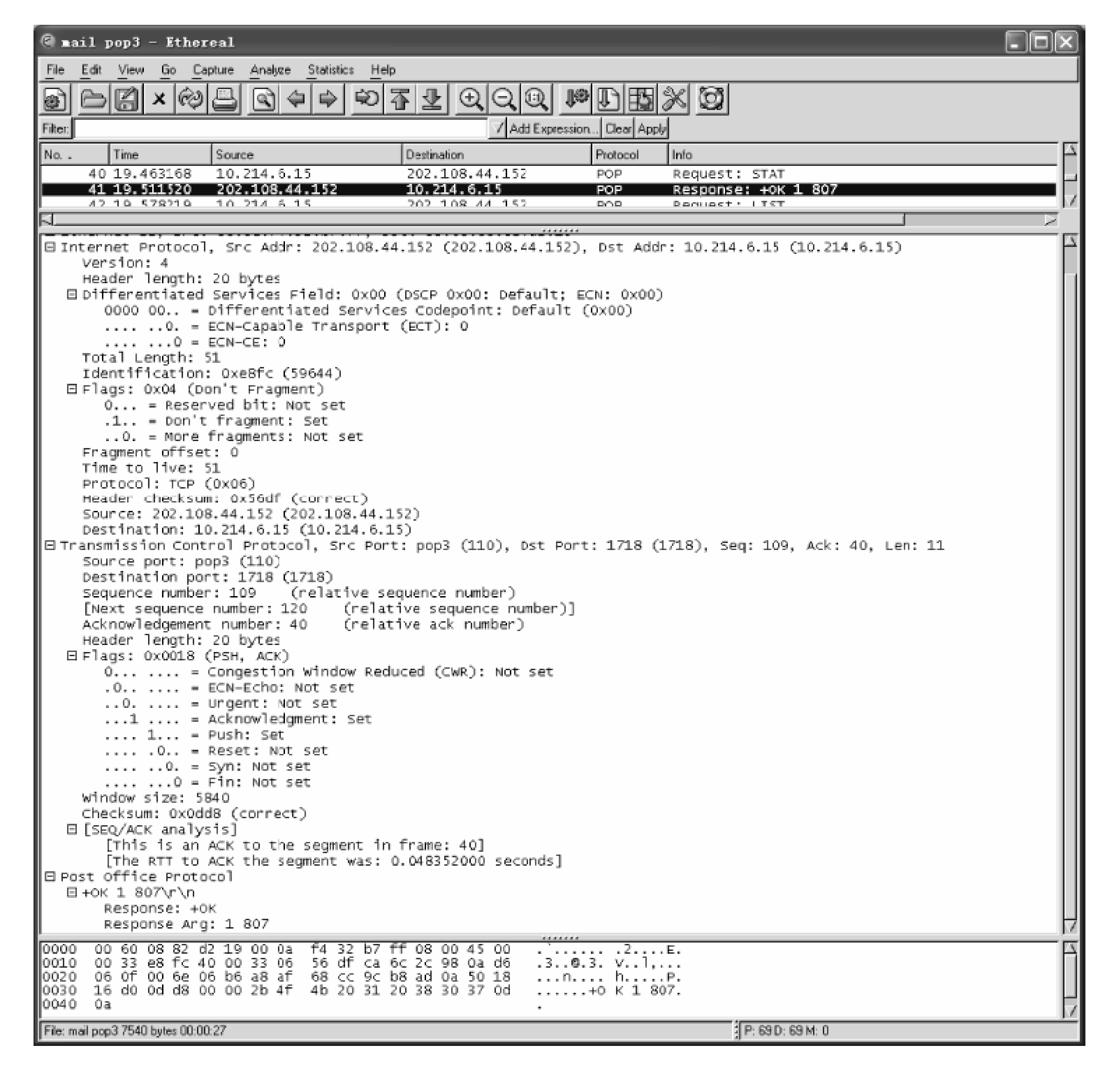

mail pop3 - Ethereal
File Edit View Go Capture Analyze Statistics Help
Filter: Add Expression... Clear Apply
No. Time Source Destination Protocol Info
40 19.463168 10.214.6.15 202.108.44.152 POP Request: STAT
41 19.511520 202.108.44.152 10.214.6.15 POP Response: +OK 1 807
Internet Protocol, Src Addr: 202.108.44.152 (202.108.44.152), Dst Addr: 10.214.6.15 (10.214.6.15)
Version: 4
Header length: 20 bytes
Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
0000 00.. = Differentiated Services Codepoint: Default (0x00)
.... ..0. = ECN-Capable Transport (ECT): 0
.... ...0 = ECN-CE: 0
Total Length: 51
Identification: 0xe8fc (59644)
Flags: 0x04 (Don't Fragment)
0... = Reserved bit: Not set
.1.. = Don't fragment: Set
..0. = More fragments: Not set
Fragment offset: 0
Time to live: 51
Protocol: TCP (0x06)
Header checksum: 0x56df (correct)
Source: 202.108.44.152 (202.108.44.152)
Destination: 10.214.6.15 (10.214.6.15)
Transmission Control Protocol, Src Port: pop3 (110), Dst Port: 1718 (1718), Seq: 109, Ack: 40, Len: 11
Source port: pop3 (110)
Destination port: 1718 (1718)
Sequence number: 109 (relative sequence number)
[Next sequence number: 120 (relative sequence number)]
Acknowledgement number: 40 (relative ack number)
Header length: 20 bytes
Flags: 0x0018 (PSH, ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 1... = Push: Set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 5840
Checksum: 0x0dd8 (correct)
[SEQ/ACK analysis]
[This is an ACK to the segment in frame: 40]
[The RTT to ACK the segment was: 0.048352000 seconds]
Post Office Protocol
+OK 1 807\r\n
Response: +OK
Response Arg: 1 807
0000 00 60 08 82 d2 19 00 0a f4 32 b7 ff 08 00 45 00
0010 00 33 e8 fc 40 00 33 06 56 df ca 6c 2c 98 0a d6
0020 06 0f 00 6e 06 b6 a8 af 68 cc 9c b8 ad 0a 50 18
0030 16 d0 0d d8 00 00 2b 4f 4b 20 31 20 38 30 37 0d
0040 0a
File: mail pop3 7540 bytes 00:00:27
P: 69 D: 69 M: 0

图 9-11

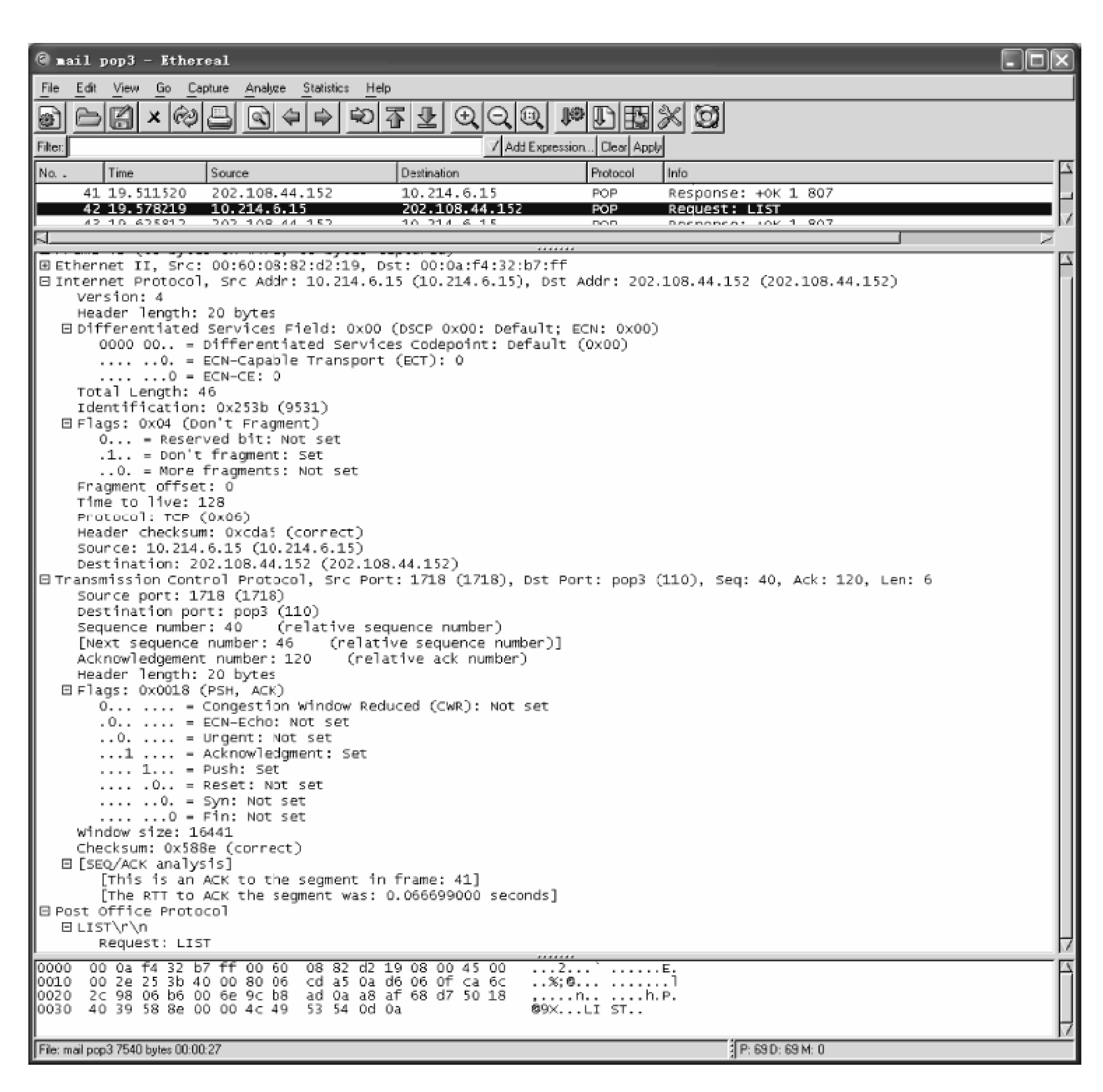

mail pop3 - Ethereal
File Edit View Go Capture Analyze Statistics Help
Filter: Add Expression... Clear Apply
No. Time Source Destination Protocol Info
41 19.511520 202.108.44.152 10.214.6.15 POP Response: +OK 1 807
42 19.578219 10.214.6.15 202.108.44.152 POP Request: LIST
Ethernet II, Src: 00:60:08:82:d2:19, Dst: 00:0a:f4:32:b7:ff
Internet Protocol, Src Addr: 10.214.6.15 (10.214.6.15), Dst Addr: 202.108.44.152 (202.108.44.152)
Version: 4
Header length: 20 bytes
Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
0000 00.. = Differentiated Services Codepoint: Default (0x00)
.... ..0. = ECN-Capable Transport (ECT): 0
.... ...0 = ECN-CE: 0
Total Length: 46
Identification: 0x253b (9531)
Flags: 0x04 (Don't Fragment)
0... = Reserved bit: Not set
.1.. = Don't fragment: Set
..0. = More fragments: Not set
Fragment offset: 0
Time to live: 128
Protocol: TCP (0x06)
Header checksum: 0xcda5 (correct)
Source: 10.214.6.15 (10.214.6.15)
Destination: 202.108.44.152 (202.108.44.152)
Transmission Control Protocol, Src Port: 1718 (1718), Dst Port: pop3 (110), Seq: 40, Ack: 120, Len: 6
Source port: 1718 (1718)
Destination port: pop3 (110)
Sequence number: 40 (relative sequence number)
[Next sequence number: 46 (relative sequence number)]
Acknowledgement number: 120 (relative ack number)
Header length: 20 bytes
Flags: 0x0018 (PSH, ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 1... = Push: Set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 16441
Checksum: 0x588e (correct)
[SEQ/ACK analysis]
[This is an ACK to the segment in frame: 41]
[The RTT to ACK the segment was: 0.066699000 seconds]
Post Office Protocol
LIST\r\n
Request: LIST
0000 00 0a f4 32 b7 ff 00 60 08 82 d2 19 08 00 45 00
0010 00 2e 25 3b 40 00 80 06 cd a5 0a d6 06 0f ca 6c
0020 2c 98 06 b6 00 6e 9c b8 ad 0a a8 af 68 d7 50 18
0030 40 39 58 8e 00 00 4c 49 53 54 0d 0a
File: mail pop3 7540 bytes 00:00:27
P: 69 D: 69 M: 0

图 9-12

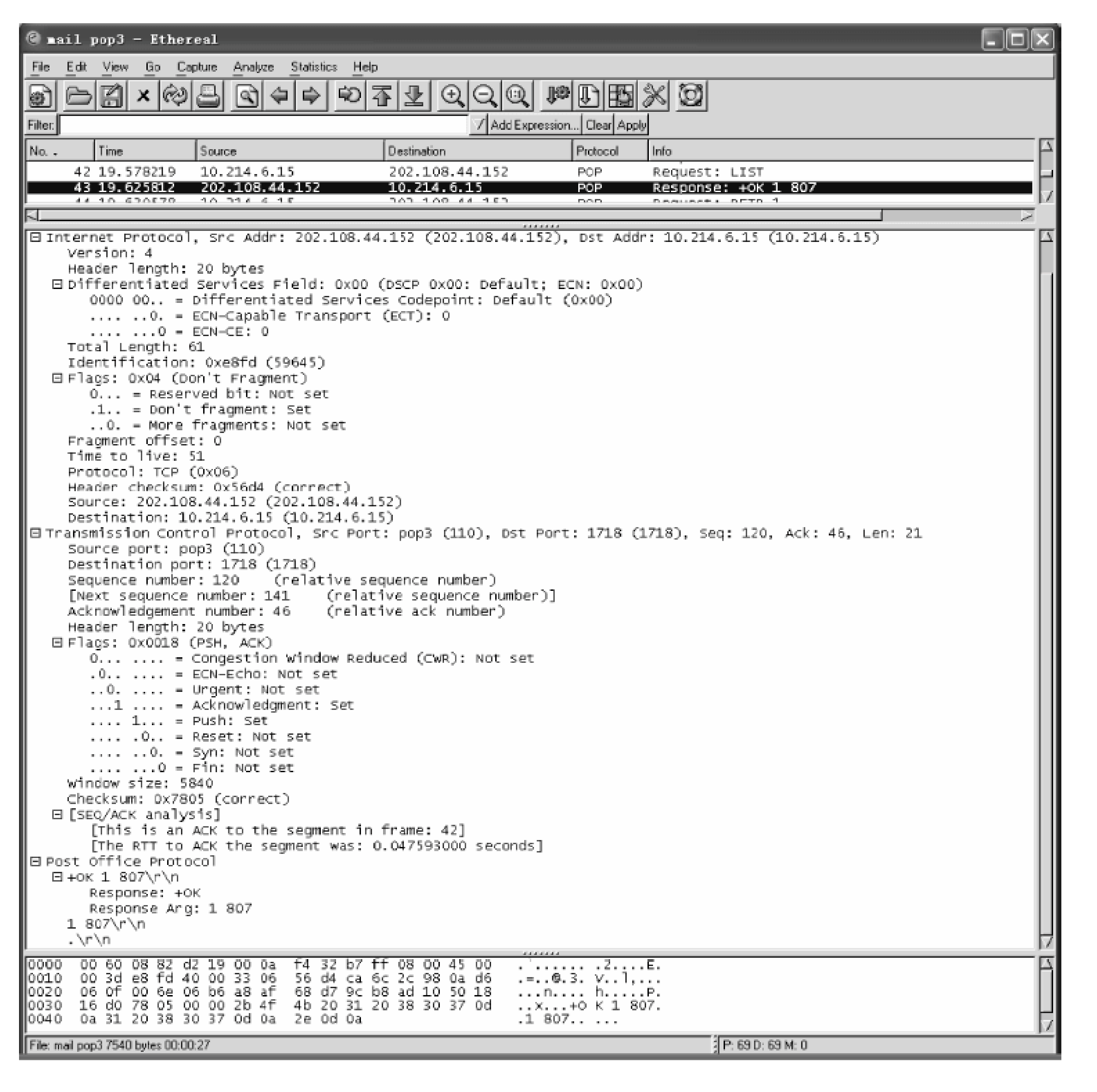



图　9-13

RETR 命令是收邮件中最重要的一条命令，它的作用是查看邮件的内容，它必须带参数进行，该命令执行之后，服务器应答的信息比较长，其中包括发件人的电子邮箱地址、发件时间、邮件主题等，这些信息统称为邮件头，紧接在邮件头之后的信息便是邮件正文。

这里的 RETR 1 表示查看第一封邮件，如图 9-14 所示。

服务器回应 RETR 请求，状态码为 OK，如图 9-15 所示。

接着服务器向客户端传输数据，如图 9-16 所示。

客户端对收到的数据进行 TCP 确认，如图 9-17 所示。

服务器端继续传输数据。可以看到当信息传送结束时，服务器端发送一个结束字符（十进制码 46 也就是十六进制码 2E，也就是“·”）和一个 CRLF 对（十六进制为 0D0A），它们的排列顺序为 CRLF-CRLF，如图 9-18 所示。

客户端用 DELE 命令是用来删除指定的邮件，如图 9-19 所示。

**注意**：DELE N 命令只是给邮件做上删除标记，只有在执行 QUIT 命令之后，邮件才会真正删除。

服务器回应 DELE 请求，状态码为 OK，如图 9-20 所示。

客户端用 QUIT 命令来结束与 POP3 服务器的会话，如图 9-21 所示。

服务器接受客户端的 QUIT 命令，返回一个信息，状态码 OK，如图 9-22 所示。

经过 TCP4 次握手，结束整个 POP3 连接，如图 9-23～图 9-26 所示。

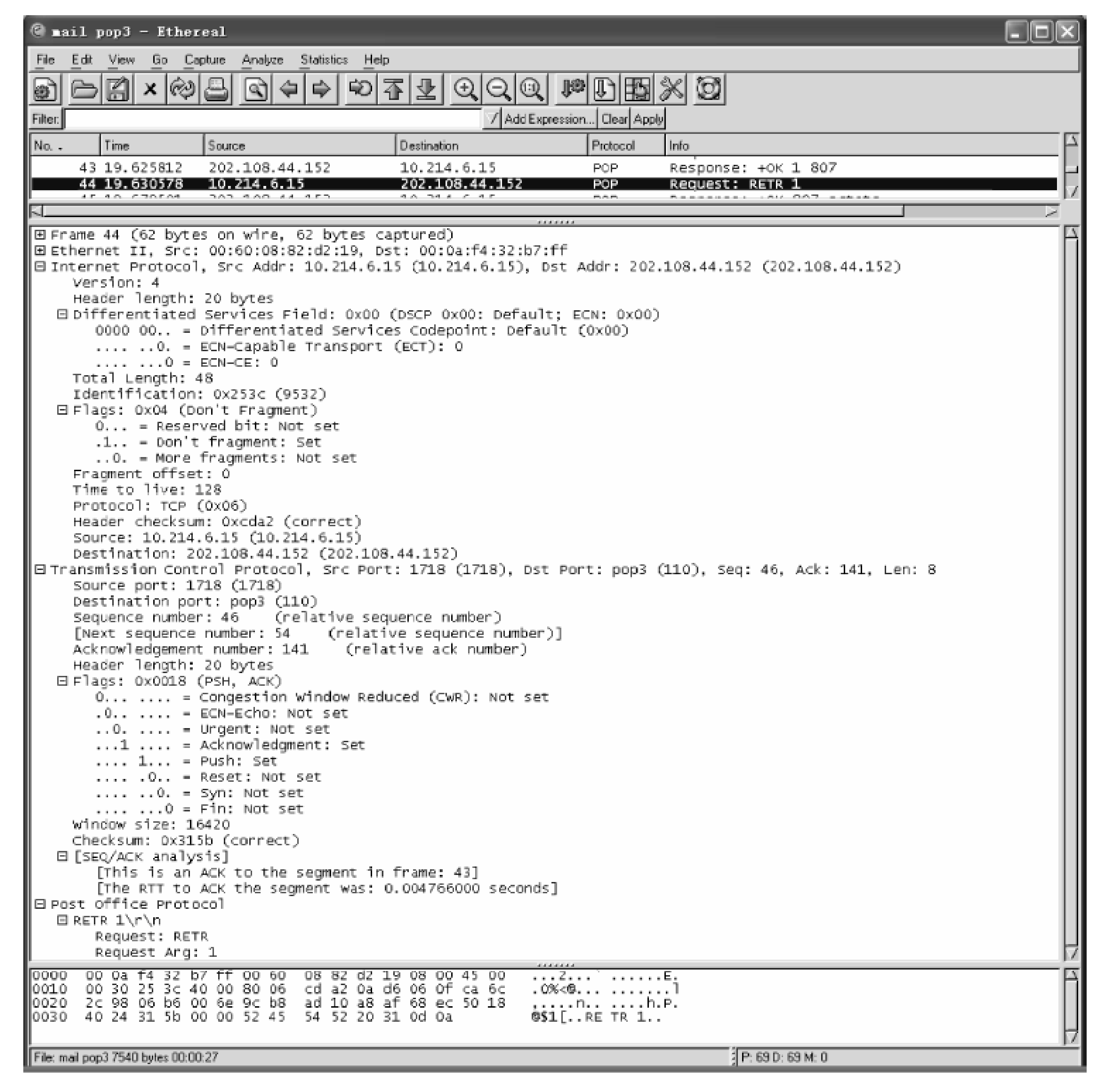

mail pop3 - Ethereal
File Edit View Go Capture Analyze Statistics Help
Filter: Add Expression... Clear Apply
No. Time Source Destination Protocol Info
43 19.625812 202.108.44.152 10.214.6.15 POP Response: +OK 1 807
44 19.630578 10.214.6.15 202.108.44.152 POP Request: RETR 1
Frame 44 (62 bytes on wire, 62 bytes captured)
Ethernet II, Src: 00:60:08:82:d2:19, Dst: 00:0a:f4:32:b7:ff
Internet Protocol, Src Addr: 10.214.6.15 (10.214.6.15), Dst Addr: 202.108.44.152 (202.108.44.152)
Version: 4
Header length: 20 bytes
Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
0000 00.. = Differentiated Services Codepoint: Default (0x00)
.... ..0. = ECN-Capable Transport (ECT): 0
.... ...0 = ECN-CE: 0
Total Length: 48
Identification: 0x253c (9532)
Flags: 0x04 (Don't Fragment)
0... = Reserved bit: Not set
.1.. = Don't fragment: Set
..0. = More fragments: Not set
Fragment offset: 0
Time to live: 128
Protocol: TCP (0x06)
Header checksum: 0xcda2 (correct)
Source: 10.214.6.15 (10.214.6.15)
Destination: 202.108.44.152 (202.108.44.152)
Transmission Control Protocol, Src Port: 1718 (1718), Dst Port: pop3 (110), Seq: 46, Ack: 141, Len: 8
Source port: 1718 (1718)
Destination port: pop3 (110)
Sequence number: 46 (relative sequence number)
[Next sequence number: 54 (relative sequence number)]
Acknowledgement number: 141 (relative ack number)
Header length: 20 bytes
Flags: 0x0018 (PSH, ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 1... = Push: Set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 16420
Checksum: 0x315b (correct)
[SEQ/ACK analysis]
[This is an ACK to the segment in frame: 43]
[The RTT to ACK the segment was: 0.004766000 seconds]
Post Office Protocol
RETR 1\r\n
Request: RETR
Request Arg: 1
0000 00 0a f4 32 b7 ff 00 60 08 82 d2 19 08 00 45 00
0010 00 30 25 3c 40 00 80 06 cd a2 0a d6 06 0f ca 6c
0020 2c 98 06 b6 00 6e 9c b8 ad 10 a8 af 68 ec 50 18
0030 40 24 31 5b 00 00 52 45 54 52 20 31 0d 0a
File: mail pop3 7540 bytes 00:00:27
P: 69 D: 69 M: 0

图 9-14

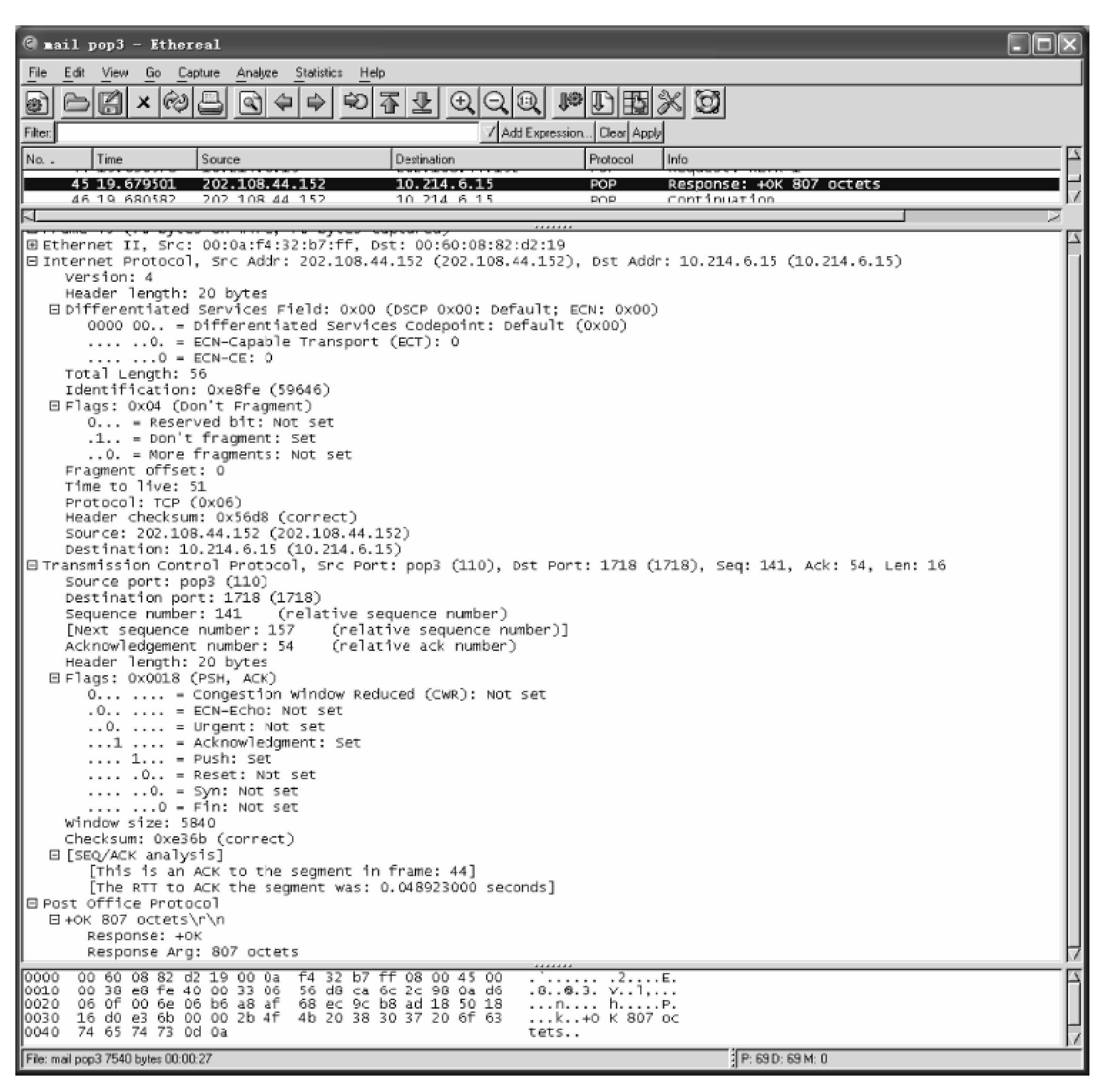

mail pop3 - Ethereal
File Edit View Go Capture Analyze Statistics Help
Filter: Add Expression... Clear Apply
No. Time Source Destination Protocol Info
45 19.679501 202.108.44.152 10.214.6.15 POP Response: +OK 807 octets
Ethernet II, Src: 00:0a:f4:32:b7:ff, Dst: 00:60:08:82:d2:19
Internet Protocol, Src Addr: 202.108.44.152 (202.108.44.152), Dst Addr: 10.214.6.15 (10.214.6.15)
Version: 4
Header length: 20 bytes
Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
0000 00.. = Differentiated Services Codepoint: Default (0x00)
.... ..0. = ECN-Capable Transport (ECT): 0
.... ...0 = ECN-CE: 0
Total Length: 56
Identification: 0xe8fe (59646)
Flags: 0x04 (Don't Fragment)
0... = Reserved bit: Not set
.1.. = Don't fragment: Set
..0. = More fragments: Not set
Fragment offset: 0
Time to live: 51
Protocol: TCP (0x06)
Header checksum: 0x56d8 (correct)
Source: 202.108.44.152 (202.108.44.152)
Destination: 10.214.6.15 (10.214.6.15)
Transmission Control Protocol, Src Port: pop3 (110), Dst Port: 1718 (1718), Seq: 141, Ack: 54, Len: 16
Source port: pop3 (110)
Destination port: 1718 (1718)
Sequence number: 141 (relative sequence number)
[Next sequence number: 157 (relative sequence number)]
Acknowledgement number: 54 (relative ack number)
Header length: 20 bytes
Flags: 0x0018 (PSH, ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 1... = Push: Set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 5840
Checksum: 0xe36b (correct)
[SEQ/ACK analysis]
[This is an ACK to the segment in frame: 44]
[The RTT to ACK the segment was: 0.048923000 seconds]
Post Office Protocol
+OK 807 octets\r\n
Response: +OK
Response Arg: 807 octets
0000 00 60 08 82 d2 19 00 0a f4 32 b7 ff 08 00 45 00
0010 00 38 e8 fe 40 00 33 06 56 d8 ca 6c 2c 98 0a d6
0020 06 0f 00 6e 06 b6 a8 af 68 ec 9c b8 ad 18 50 18
0030 16 d0 e3 6b 00 00 2b 4f 4b 20 38 30 37 20 6f 63
0040 74 65 74 73 0d 0a
File: mail pop3 7540 bytes 00:00:27
P: 69 D: 69 M: 0

图 9-15

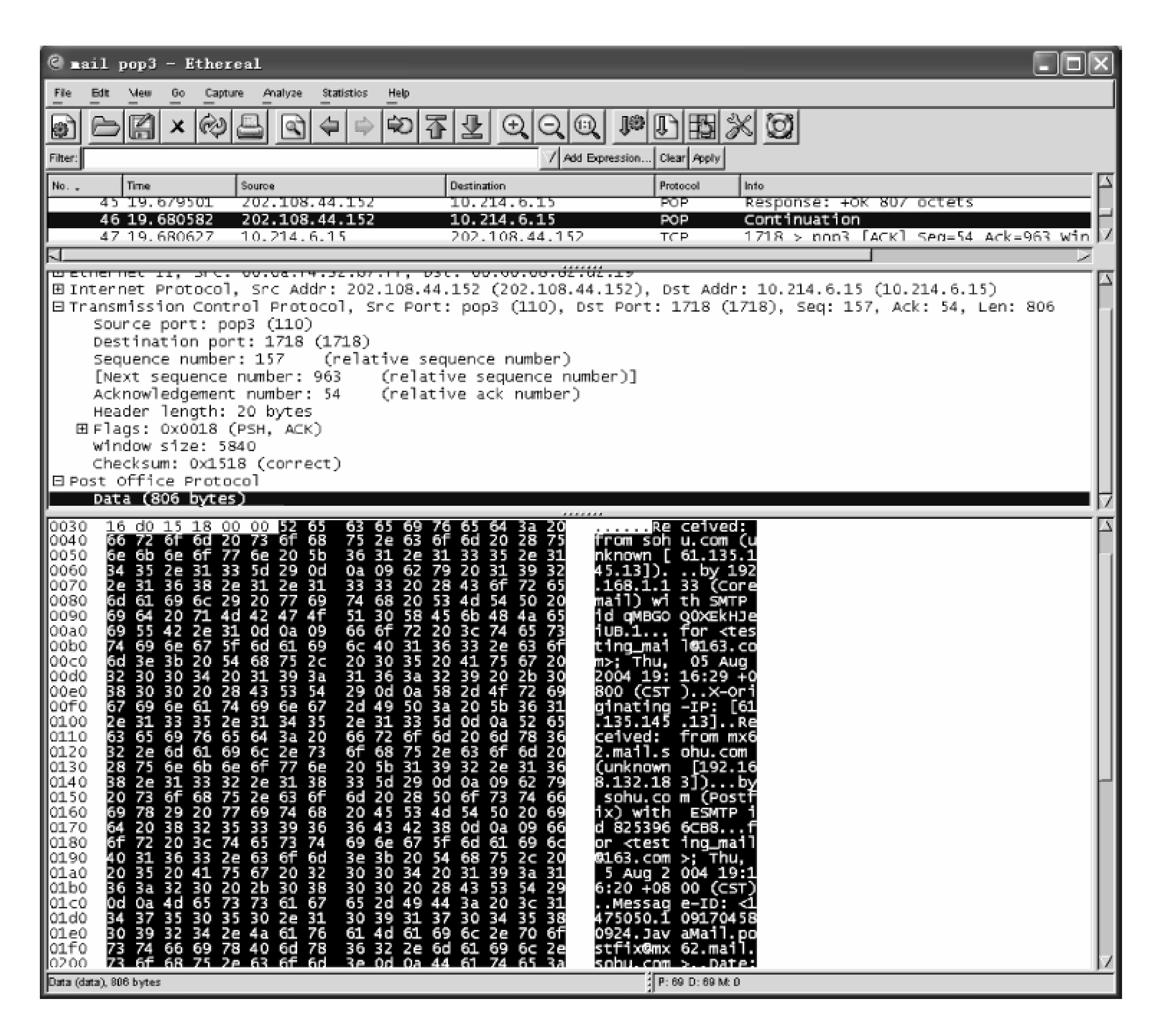

mail pop3 - Ethereal
File Edit View Go Capture Analyze Statistics Help
46 19.680582 202.108.44.152 10.214.6.15 POP Continuation
Internet Protocol, Src Addr: 202.108.44.152 (202.108.44.152), Dst Addr: 10.214.6.15 (10.214.6.15)
Transmission Control Protocol, Src Port: pop3 (110), Dst Port: 1718 (1718), Seq: 157, Ack: 54, Len: 806
Source port: pop3 (110)
Destination port: 1718 (1718)
Sequence number: 157 (relative sequence number)
[Next sequence number: 963 (relative sequence number)]
Acknowledgement number: 54 (relative ack number)
Header length: 20 bytes
Flags: 0x0018 (PSH, ACK)
Window size: 5840
Checksum: 0x1518 (correct)
Post Office Protocol
Data (806 bytes)
Data (data), 806 bytes

图　9-16

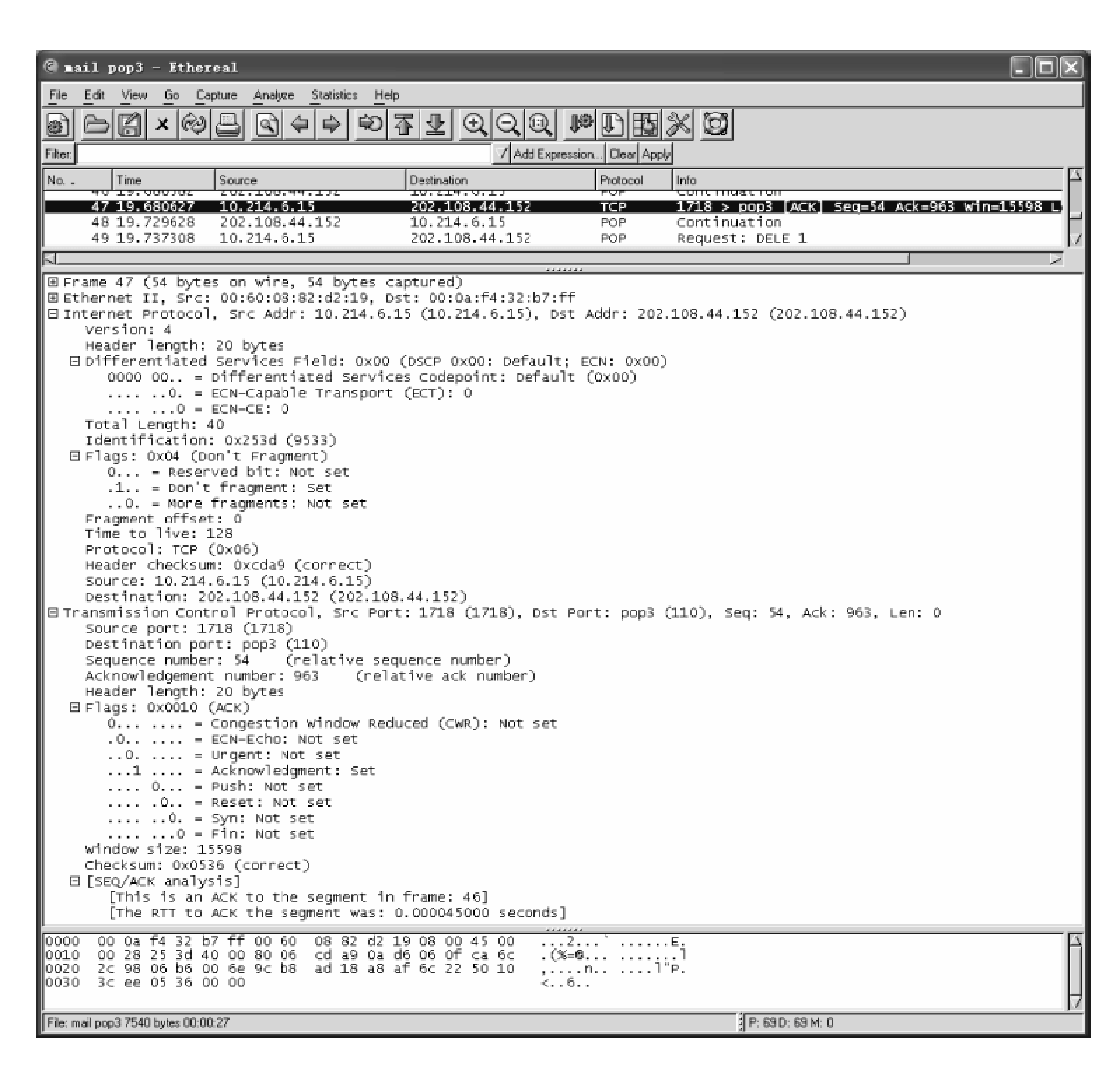

mail pop3 - Ethereal
File Edit View Go Capture Analyze Statistics Help
47 19.680627 10.214.6.15 202.108.44.152 TCP 1718 > pop3 [ACK] Seq=54 Ack=963 Win=15598
48 19.729628 202.108.44.152 10.214.6.15 POP Continuation
49 19.737308 10.214.6.15 202.108.44.152 POP Request: DELE 1
Frame 47 (54 bytes on wire, 54 bytes captured)
Ethernet II, Src: 00:60:08:82:d2:19, Dst: 00:0a:f4:32:b7:ff
Internet Protocol, Src Addr: 10.214.6.15 (10.214.6.15), Dst Addr: 202.108.44.152 (202.108.44.152)
Version: 4
Header length: 20 bytes
Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
Total Length: 40
Identification: 0x253d (9533)
Flags: 0x04 (Don't Fragment)
Fragment offset: 0
Time to live: 128
Protocol: TCP (0x06)
Header checksum: 0xcda9 (correct)
Source: 10.214.6.15 (10.214.6.15)
Destination: 202.108.44.152 (202.108.44.152)
Transmission Control Protocol, Src Port: 1718 (1718), Dst Port: pop3 (110), Seq: 54, Ack: 963, Len: 0
Source port: 1718 (1718)
Destination port: pop3 (110)
Sequence number: 54 (relative sequence number)
Acknowledgement number: 963 (relative ack number)
Header length: 20 bytes
Flags: 0x0010 (ACK)
Window size: 15598
Checksum: 0x0536 (correct)
[SEQ/ACK analysis]
[This is an ACK to the segment in frame: 46]
[The RTT to ACK the segment was: 0.000045000 seconds]

图　9-17

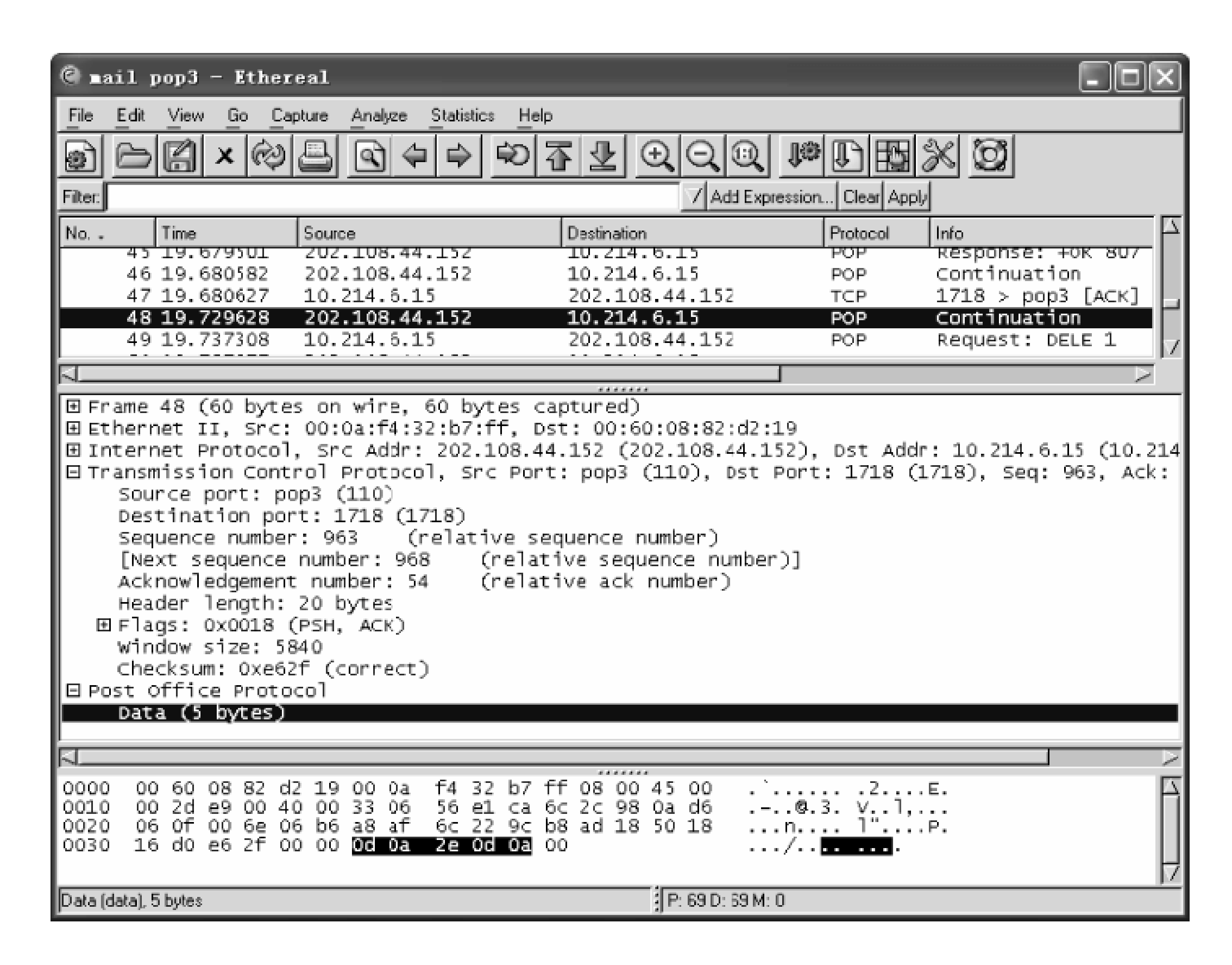

mail pop3 - Ethereal
File Edit View Go Capture Analyze Statistics Help
Filter: Add Expression... Clear Apply
No. Time Source Destination Protocol Info
46 19.680582 202.108.44.152 10.214.6.15 POP Continuation
47 19.680627 10.214.6.15 202.108.44.152 TCP 1718 > pop3 [ACK]
48 19.729628 202.108.44.152 10.214.6.15 POP Continuation
49 19.737308 10.214.6.15 202.108.44.152 POP Request: DELE 1
Frame 48 (60 bytes on wire, 60 bytes captured)
Ethernet II, Src: 00:0a:f4:32:b7:ff, Dst: 00:60:08:82:d2:19
Internet Protocol, Src Addr: 202.108.44.152 (202.108.44.152), Dst Addr: 10.214.6.15 (10.214
Transmission Control Protocol, Src Port: pop3 (110), Dst Port: 1718 (1718), Seq: 963, Ack:
Source port: pop3 (110)
Destination port: 1718 (1718)
Sequence number: 963 (relative sequence number)
[Next sequence number: 968 (relative sequence number)]
Acknowledgement number: 54 (relative ack number)
Header length: 20 bytes
Flags: 0x0018 (PSH, ACK)
Window size: 5840
Checksum: 0xe62f (correct)
Post Office Protocol
Data (5 bytes)
0000 00 60 08 82 d2 19 00 0a f4 32 b7 ff 08 00 45 00
0010 00 2d e9 00 40 00 33 06 56 e1 ca 6c 2c 98 0a d6
0020 06 0f 00 6e 06 b6 a8 af 6c 22 9c b8 ad 18 50 18
0030 16 d0 e6 2f 00 00 0d 0a 2e 0d 0a 00
Data (data), 5 bytes
P: 69 D: 69 M: 0

图 9-18

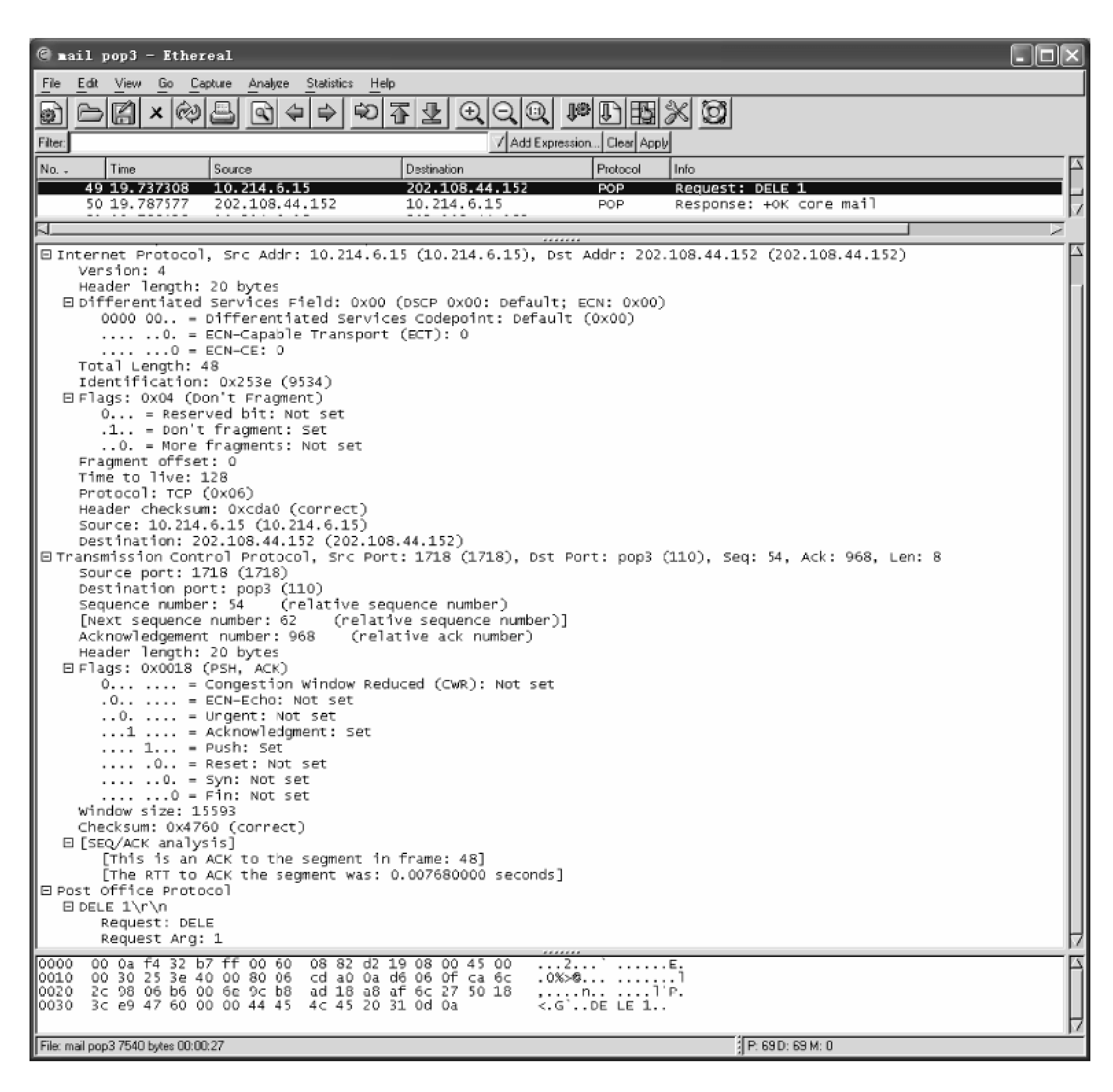

mail pop3 - Ethereal
File Edit View Go Capture Analyze Statistics Help
Filter: Add Expression... Clear Apply
No. Time Source Destination Protocol Info
49 19.737308 10.214.6.15 202.108.44.152 POP Request: DELE 1
50 19.787577 202.108.44.152 10.214.6.15 POP Response: +OK core mail
Internet Protocol, Src Addr: 10.214.6.15 (10.214.6.15), Dst Addr: 202.108.44.152 (202.108.44.152)
Version: 4
Header length: 20 bytes
Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
0000 00.. = Differentiated Services Codepoint: Default (0x00)
.... ..0. = ECN-Capable Transport (ECT): 0
.... ...0 = ECN-CE: 0
Total Length: 48
Identification: 0x253e (9534)
Flags: 0x04 (Don't Fragment)
0... = Reserved bit: Not set
.1.. = Don't fragment: Set
..0. = More fragments: Not set
Fragment offset: 0
Time to live: 128
Protocol: TCP (0x06)
Header checksum: 0xcda0 (correct)
Source: 10.214.6.15 (10.214.6.15)
Destination: 202.108.44.152 (202.108.44.152)
Transmission Control Protocol, Src Port: 1718 (1718), Dst Port: pop3 (110), Seq: 54, Ack: 968, Len: 8
Source port: 1718 (1718)
Destination port: pop3 (110)
Sequence number: 54 (relative sequence number)
[Next sequence number: 62 (relative sequence number)]
Acknowledgement number: 968 (relative ack number)
Header length: 20 bytes
Flags: 0x0018 (PSH, ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 1... = Push: Set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 15593
Checksum: 0x4760 (correct)
[SEQ/ACK analysis]
[This is an ACK to the segment in frame: 48]
[The RTT to ACK the segment was: 0.007680000 seconds]
Post Office Protocol
DELE 1\r\n
Request: DELE
Request Arg: 1
0000 00 0a f4 32 b7 ff 00 60 08 82 d2 19 08 00 45 00
0010 00 30 25 3e 40 00 80 06 cd a0 0a d6 06 0f ca 6c
0020 2c 98 06 b6 00 6e 9c b8 ad 18 a8 af 6c 27 50 18
0030 3c e9 47 60 00 00 44 45 4c 45 20 31 0d 0a
<.G`..DE LE 1..
File: mail pop3 7540 bytes 00:00:27
P: 69 D: 69 M: 0

图 9-19

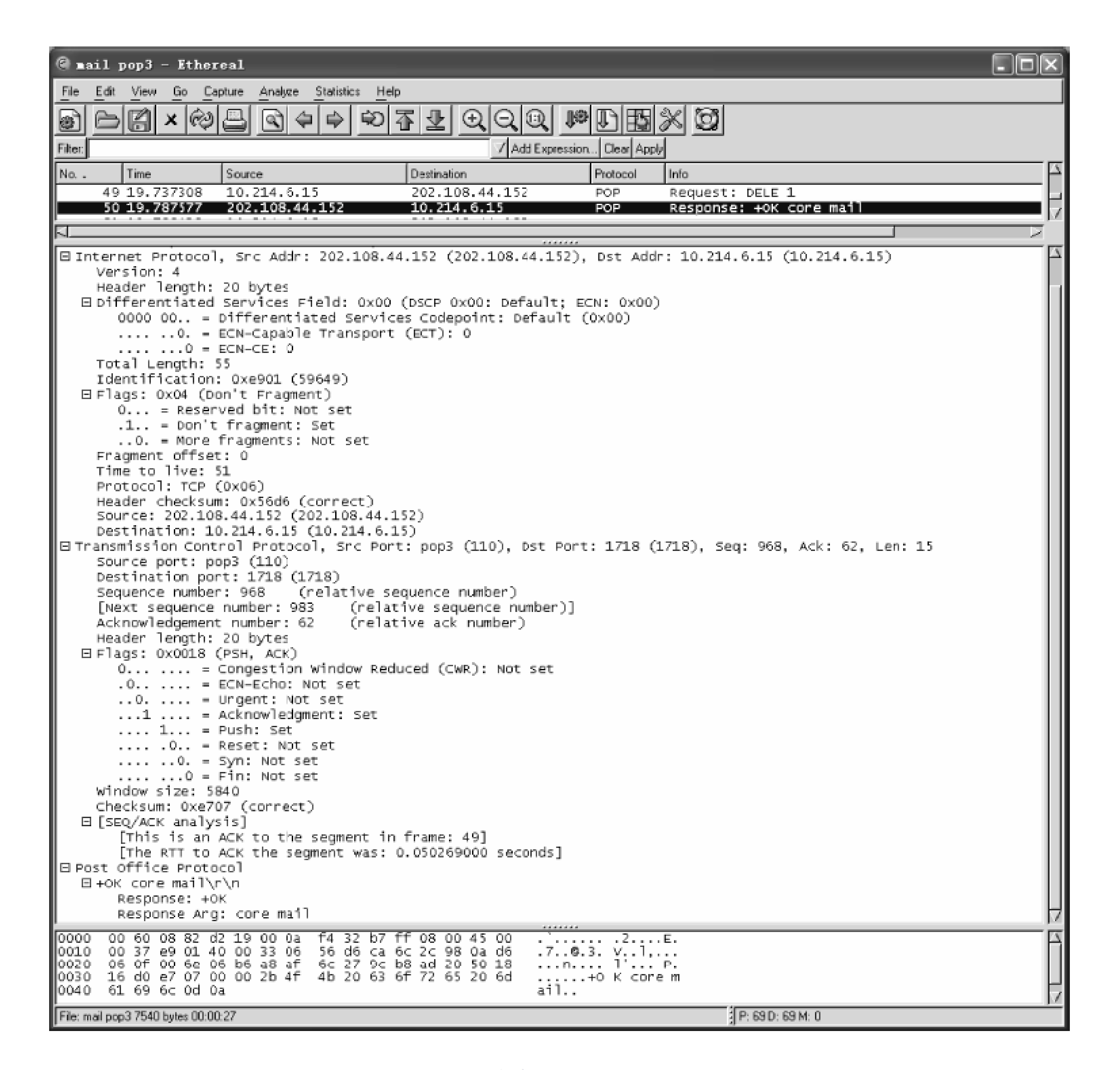

mail pop3 - Ethereal
File Edit View Go Capture Analyze Statistics Help
Filter:
Add Expression... Clear Apply
No. . Time Source Destination Protocol Info
49 19.737308 10.214.6.15 202.108.44.152 POP Request: DELE 1
50 19.787377 202.108.44.152 10.214.6.15 POP Response: +OK core mail
Internet Protocol, Src Addr: 202.108.44.152 (202.108.44.152), Dst Addr: 10.214.6.15 (10.214.6.15)
Version: 4
Header length: 20 bytes
Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
0000 00.. = Differentiated Services Codepoint: Default (0x00)
.... ..0. = ECN-Capable Transport (ECT): 0
.... ...0 = ECN-CE: 0
Total Length: 55
Identification: 0xe901 (59649)
Flags: 0x04 (Don't Fragment)
0... = Reserved bit: Not set
.1.. = Don't fragment: Set
..0. = More fragments: Not set
Fragment offset: 0
Time to live: 51
Protocol: TCP (0x06)
Header checksum: 0x56d6 (correct)
Source: 202.108.44.152 (202.108.44.152)
Destination: 10.214.6.15 (10.214.6.15)
Transmission Control Protocol, Src Port: pop3 (110), Dst Port: 1718 (1718), Seq: 968, Ack: 62, Len: 15
Source port: pop3 (110)
Destination port: 1718 (1718)
Sequence number: 968 (relative sequence number)
[Next sequence number: 983 (relative sequence number)]
Acknowledgement number: 62 (relative ack number)
Header length: 20 bytes
Flags: 0x0018 (PSH, ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 1... = Push: Set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 5840
Checksum: 0xe707 (correct)
[SEQ/ACK analysis]
[This is an ACK to the segment in frame: 49]
[The RTT to ACK the segment was: 0.050269000 seconds]
Post Office Protocol
+OK core mail\r\n
Response: +OK
Response Arg: core mail
0000 00 60 08 82 d2 19 00 0a f4 32 b7 ff 08 00 45 00
0010 00 37 e9 01 40 00 33 06 56 d6 ca 6c 2c 98 0a d6
0020 06 0f 00 6e 06 b6 a8 af 6c 27 9c b8 ad 20 50 18
0030 16 d0 e7 07 00 00 2b 4f 4b 20 63 6f 72 65 20 6d
0040 61 69 6c 0d 0a
File: mail pop3 7540 bytes 00:00:27
P: 69 D: 69 M: 0

图　9-20

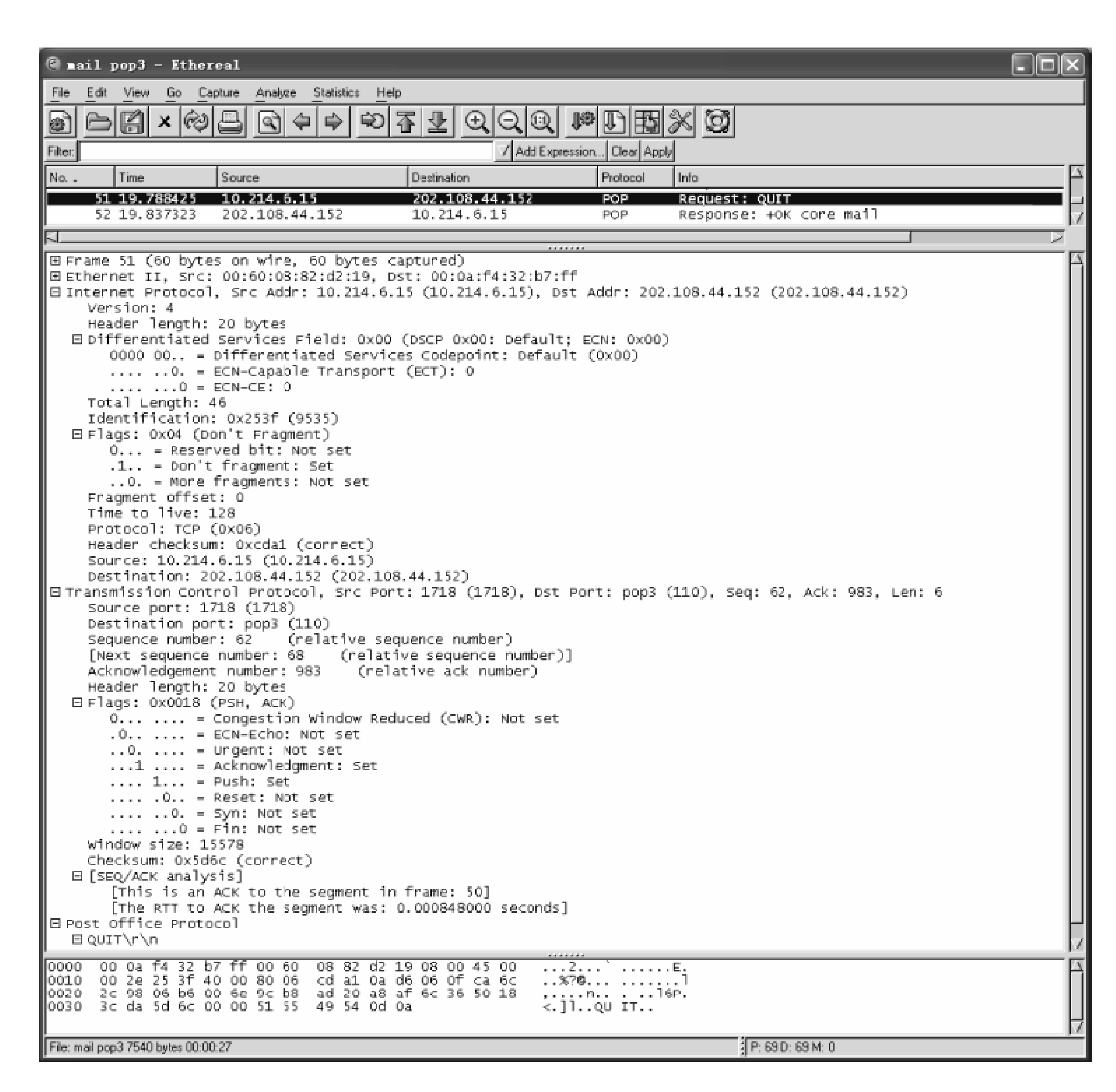

mail pop3 - Ethereal
File Edit View Go Capture Analyze Statistics Help
Filter:
Add Expression... Clear Apply
No. . Time Source Destination Protocol Info
51 19.788425 10.214.6.15 202.108.44.152 POP Request: QUIT
52 19.837323 202.108.44.152 10.214.6.15 POP Response: +OK core mail
Frame 51 (60 bytes on wire, 60 bytes captured)
Ethernet II, Src: 00:60:08:82:d2:19, Dst: 00:0a:f4:32:b7:ff
Internet Protocol, Src Addr: 10.214.6.15 (10.214.6.15), Dst Addr: 202.108.44.152 (202.108.44.152)
Version: 4
Header length: 20 bytes
Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
0000 00.. = Differentiated Services Codepoint: Default (0x00)
.... ..0. = ECN-Capable Transport (ECT): 0
.... ...0 = ECN-CE: 0
Total Length: 46
Identification: 0x253f (9535)
Flags: 0x04 (Don't Fragment)
0... = Reserved bit: Not set
.1.. = Don't fragment: Set
..0. = More fragments: Not set
Fragment offset: 0
Time to live: 128
Protocol: TCP (0x06)
Header checksum: 0xcda1 (correct)
Source: 10.214.6.15 (10.214.6.15)
Destination: 202.108.44.152 (202.108.44.152)
Transmission Control Protocol, Src Port: 1718 (1718), Dst Port: pop3 (110), Seq: 62, Ack: 983, Len: 6
Source port: 1718 (1718)
Destination port: pop3 (110)
Sequence number: 62 (relative sequence number)
[Next sequence number: 68 (relative sequence number)]
Acknowledgement number: 983 (relative ack number)
Header length: 20 bytes
Flags: 0x0018 (PSH, ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 1... = Push: Set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 15578
Checksum: 0x5d6c (correct)
[SEQ/ACK analysis]
[This is an ACK to the segment in frame: 50]
[The RTT to ACK the segment was: 0.000848000 seconds]
Post Office Protocol
QUIT\r\n
0000 00 0a f4 32 b7 ff 00 60 08 82 d2 19 08 00 45 00
0010 00 2e 25 3f 40 00 80 06 cd a1 0a d6 06 0f ca 6c
0020 2c 98 06 b6 00 6e 9c b8 ad 20 a8 af 6c 36 50 18
0030 3c da 5d 6c 00 00 51 55 49 54 0d 0a
File: mail pop3 7540 bytes 00:00:27
P: 69 D: 69 M: 0

图　9-21

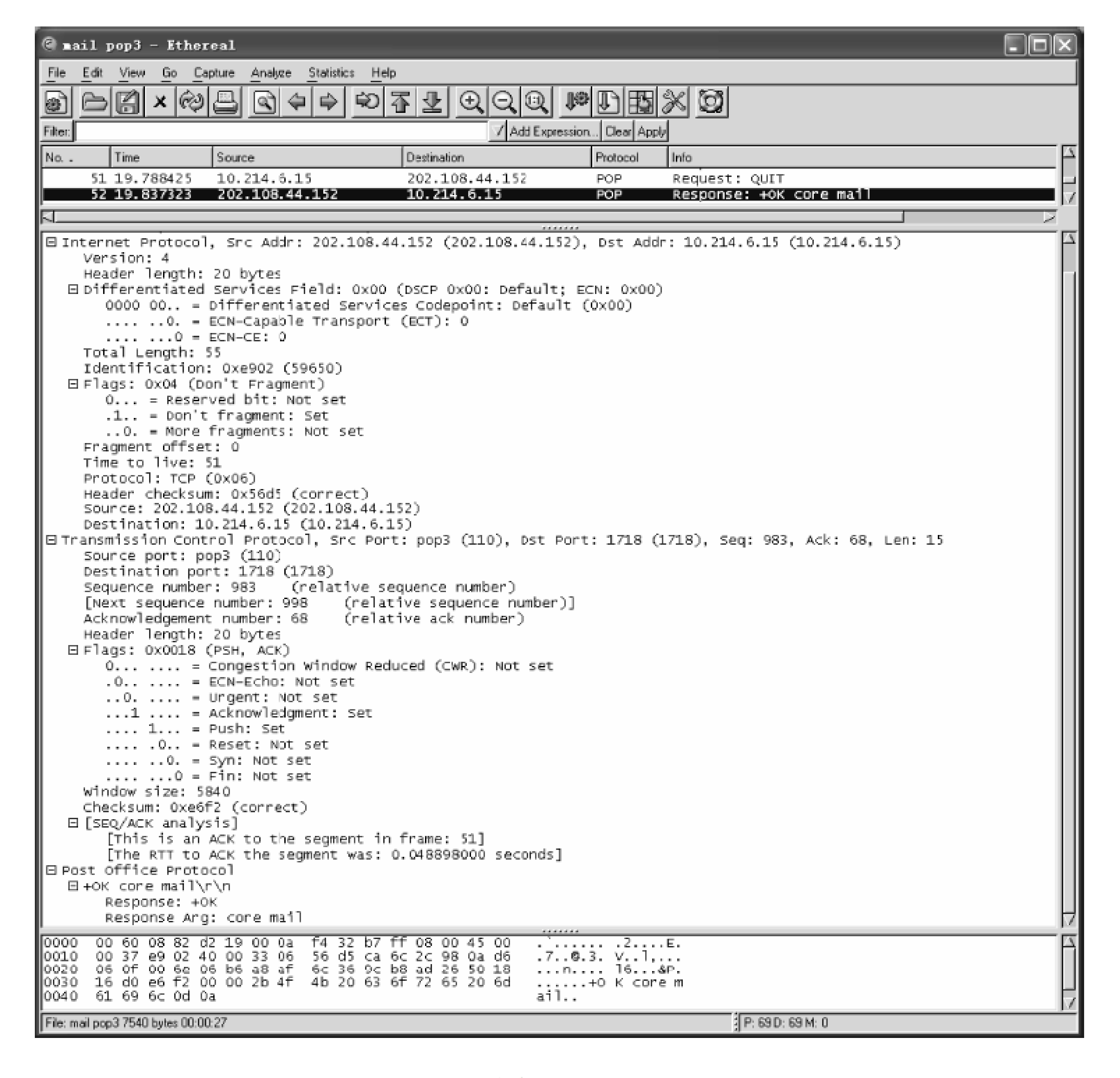

mail pop3 - Ethereal
File Edit View Go Capture Analyze Statistics Help
Filter: Add Expression... Clear Apply
No. Time Source Destination Protocol Info
51 19.788425 10.214.6.15 202.108.44.152 POP Request: QUIT
52 19.837323 202.108.44.152 10.214.6.15 POP Response: +OK core mail
Internet Protocol, Src Addr: 202.108.44.152 (202.108.44.152), Dst Addr: 10.214.6.15 (10.214.6.15)
Version: 4
Header length: 20 bytes
Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
0000 00.. = Differentiated Services Codepoint: Default (0x00)
.... ..0. = ECN-Capable Transport (ECT): 0
.... ...0 = ECN-CE: 0
Total Length: 55
Identification: 0xe902 (59650)
Flags: 0x04 (Don't Fragment)
0... = Reserved bit: Not set
.1.. = Don't fragment: Set
..0. = More fragments: Not set
Fragment offset: 0
Time to live: 51
Protocol: TCP (0x06)
Header checksum: 0x56d5 (correct)
Source: 202.108.44.152 (202.108.44.152)
Destination: 10.214.6.15 (10.214.6.15)
Transmission Control Protocol, Src Port: pop3 (110), Dst Port: 1718 (1718), Seq: 983, Ack: 68, Len: 15
Source port: pop3 (110)
Destination port: 1718 (1718)
Sequence number: 983 (relative sequence number)
[Next sequence number: 998 (relative sequence number)]
Acknowledgement number: 68 (relative ack number)
Header length: 20 bytes
Flags: 0x0018 (PSH, ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 1... = Push: Set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 5840
Checksum: 0xe6f2 (correct)
[SEQ/ACK analysis]
[This is an ACK to the segment in frame: 51]
[The RTT to ACK the segment was: 0.048898000 seconds]
Post Office Protocol
+OK core mail\r\n
Response: +OK
Response Arg: core mail
0000 00 60 08 82 d2 19 00 0a f4 32 b7 ff 08 00 45 00
0010 00 37 e9 02 40 00 33 06 56 d5 ca 6c 2c 98 0a d6
0020 06 0f 00 6e 06 b6 a8 af 6c 36 9c b8 ad 26 50 18
0030 16 d0 e6 f2 00 00 2b 4f 4b 20 63 6f 72 65 20 6d
0040 61 69 6c 0d 0a
File: mail pop3 7540 bytes 00:00:27
P: 69 D: 69 M: 0

图 9-22

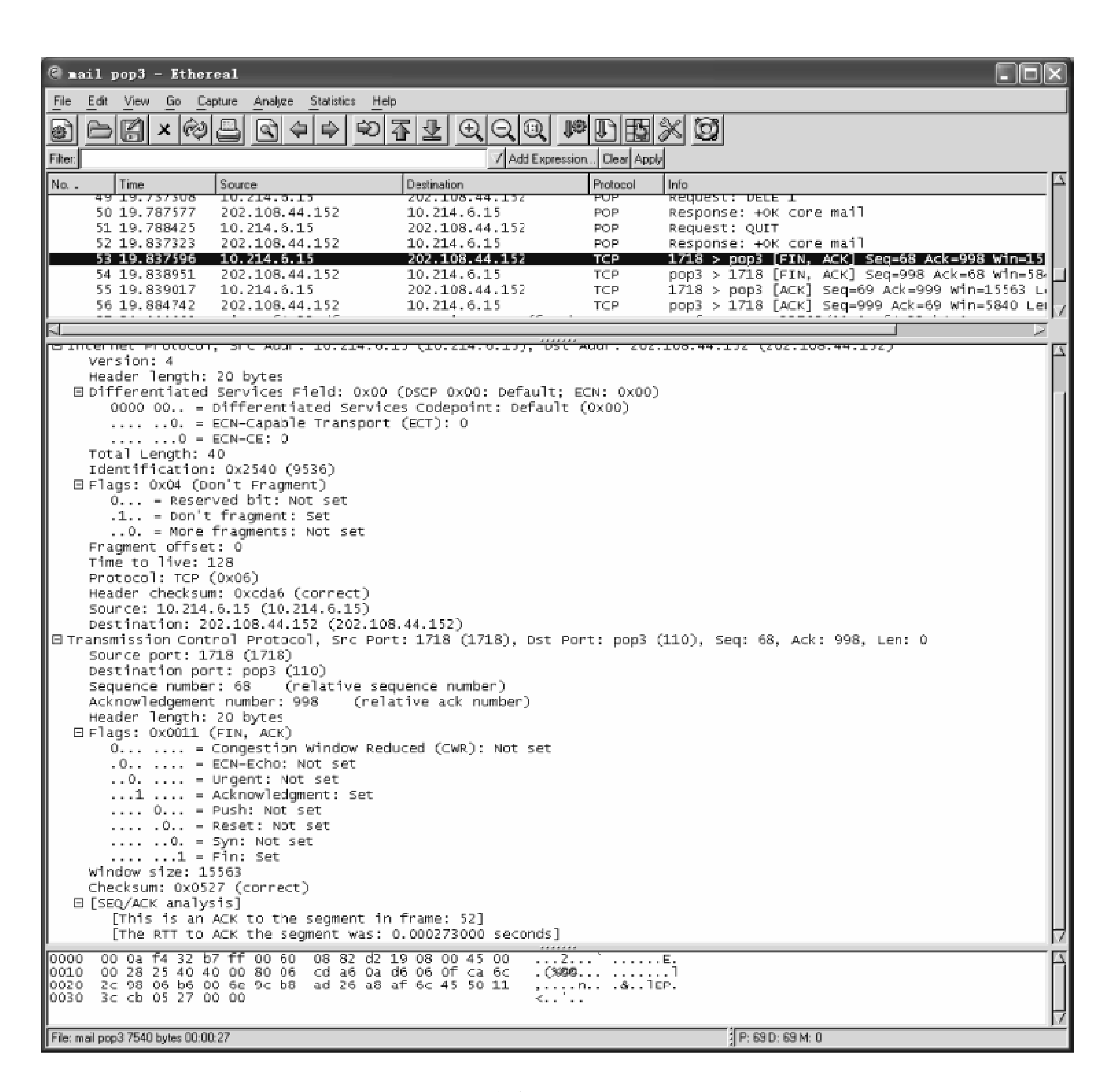

mail pop3 - Ethereal
File Edit View Go Capture Analyze Statistics Help
Filter: Add Expression... Clear Apply
No. Time Source Destination Protocol Info
50 19.787577 202.108.44.152 10.214.6.15 POP Response: +OK core mail
51 19.788425 10.214.6.15 202.108.44.152 POP Request: QUIT
52 19.837323 202.108.44.152 10.214.6.15 POP Response: +OK core mail
53 19.837596 10.214.6.15 202.108.44.152 TCP 1718 > pop3 [FIN, ACK] Seq=68 Ack=998 Win=15
54 19.838951 202.108.44.152 10.214.6.15 TCP pop3 > 1718 [FIN, ACK] Seq=998 Ack=68 Win=58
55 19.839017 10.214.6.15 202.108.44.152 TCP 1718 > pop3 [ACK] Seq=69 Ack=999 Win=15563
56 19.884742 202.108.44.152 10.214.6.15 TCP pop3 > 1718 [ACK] Seq=999 Ack=69 Win=5840
Version: 4
Header length: 20 bytes
Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
0000 00.. = Differentiated Services Codepoint: Default (0x00)
.... ..0. = ECN-Capable Transport (ECT): 0
.... ...0 = ECN-CE: 0
Total Length: 40
Identification: 0x2540 (9536)
Flags: 0x04 (Don't Fragment)
0... = Reserved bit: Not set
.1.. = Don't fragment: Set
..0. = More fragments: Not set
Fragment offset: 0
Time to live: 128
Protocol: TCP (0x06)
Header checksum: 0xcda6 (correct)
Source: 10.214.6.15 (10.214.6.15)
Destination: 202.108.44.152 (202.108.44.152)
Transmission Control Protocol, Src Port: 1718 (1718), Dst Port: pop3 (110), Seq: 68, Ack: 998, Len: 0
Source port: 1718 (1718)
Destination port: pop3 (110)
Sequence number: 68 (relative sequence number)
Acknowledgement number: 998 (relative ack number)
Header length: 20 bytes
Flags: 0x0011 (FIN, ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 0... = Push: Not set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...1 = Fin: Set
Window size: 15563
Checksum: 0x0527 (correct)
[SEQ/ACK analysis]
[This is an ACK to the segment in frame: 52]
[The RTT to ACK the segment was: 0.000273000 seconds]
0000 00 0a f4 32 b7 ff 00 60 08 82 d2 19 08 00 45 00
0010 00 28 25 40 40 00 80 06 cd a6 0a d6 06 0f ca 6c
0020 2c 98 06 b6 00 6e 9c b8 ad 26 a8 af 6c 45 50 11
0030 3c cb 05 27 00 00
File: mail pop3 7540 bytes 00:00:27
P: 69 D: 69 M: 0

图 9-23

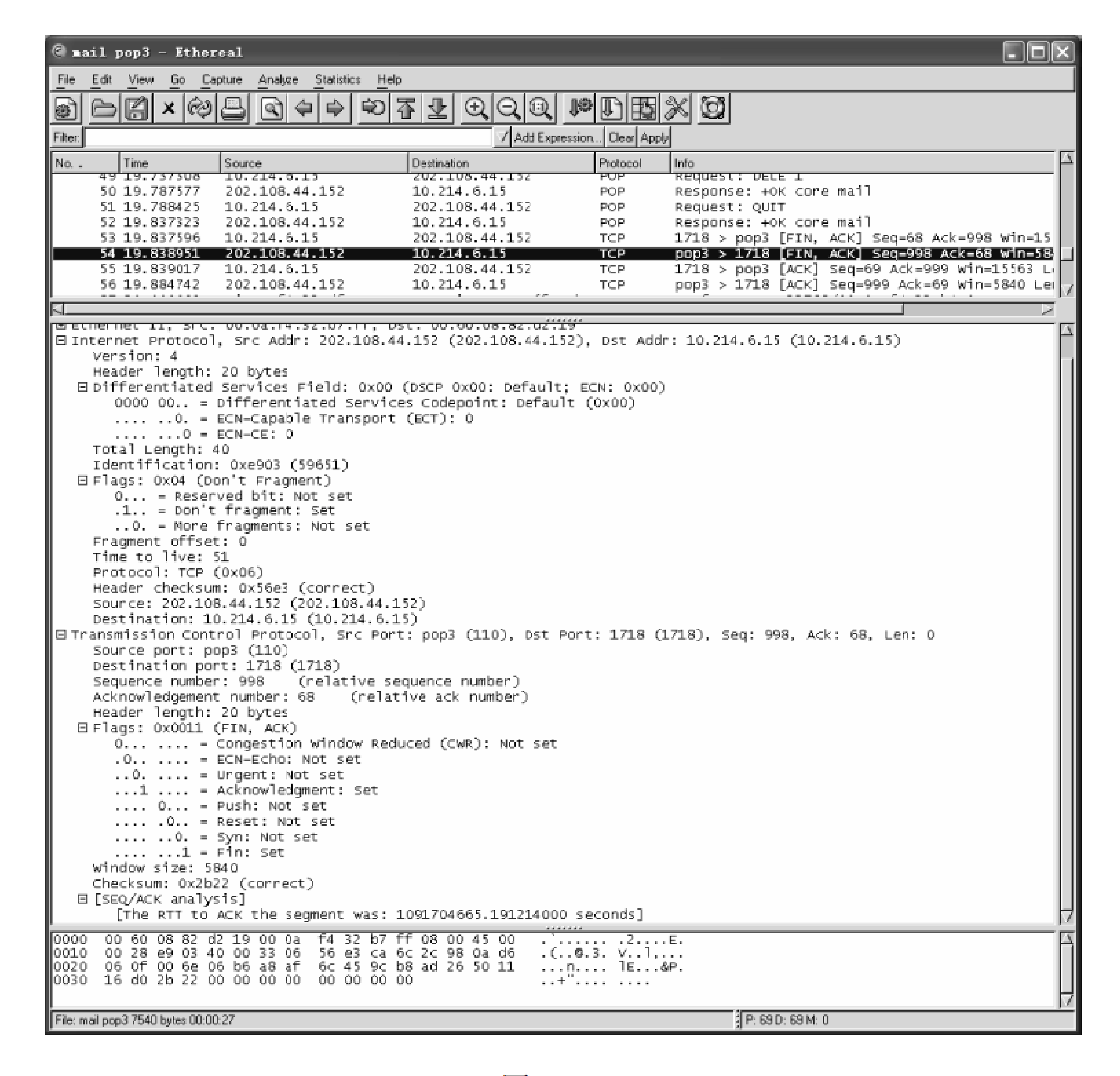


图　9-24

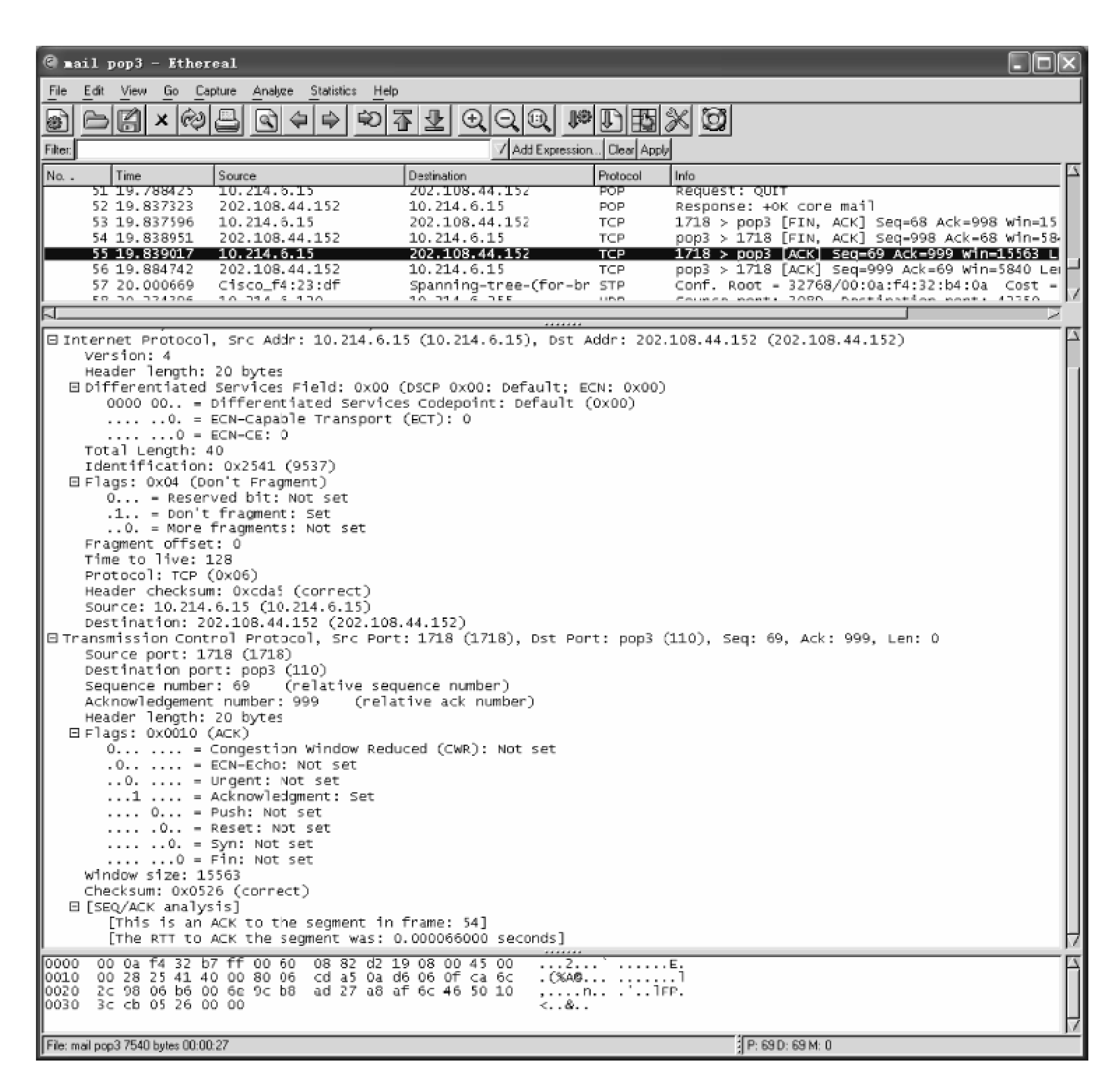


图　9-25

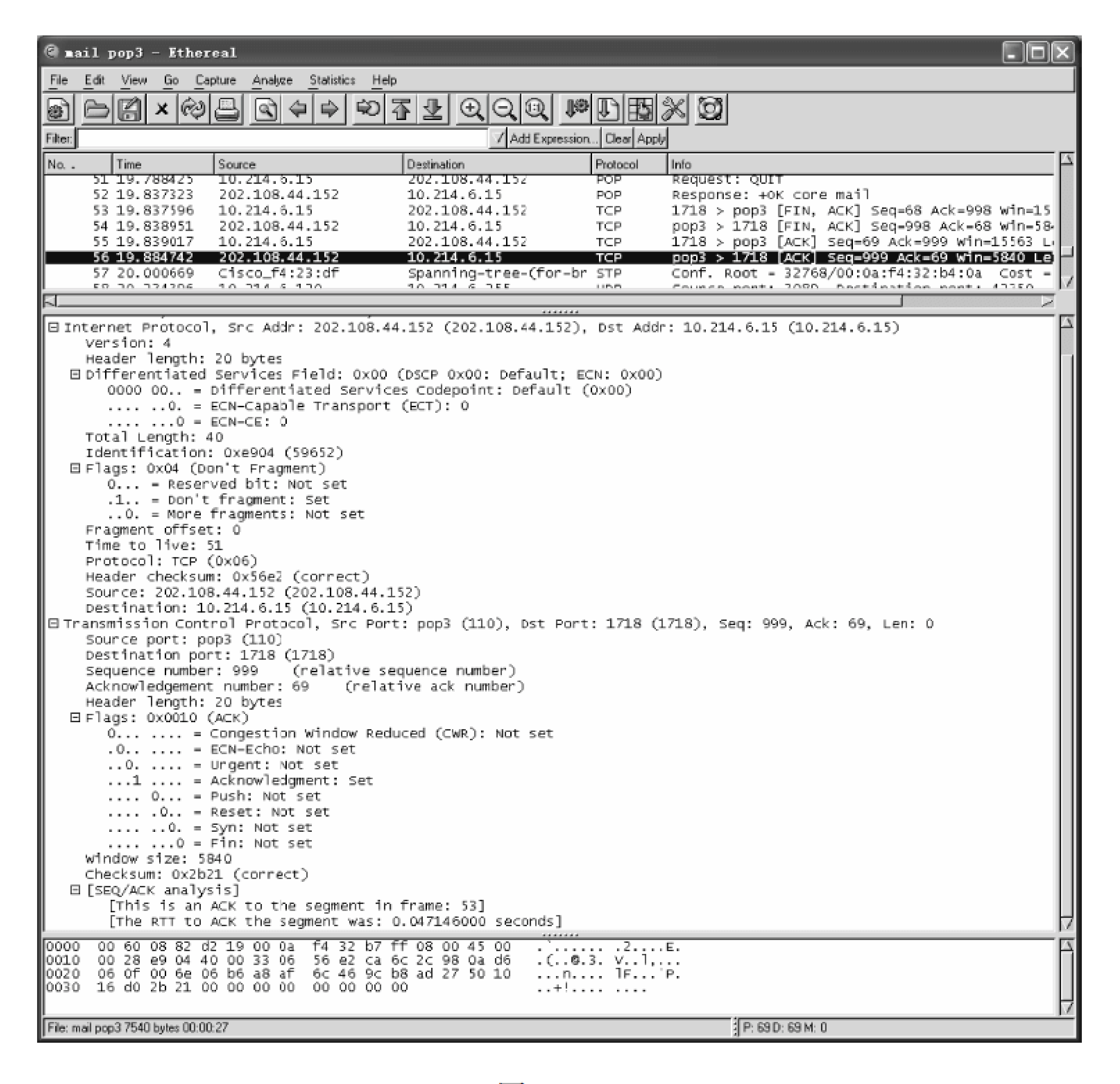



图 9-26

## 9.2 SMTP 协议

发送电子邮件用到的就是 SMTP 协议。当发送一封邮件时，假设用户的邮件地址是 zhangsan@domain1.edu.cn，发送到 lisi@domain2.edu.cn，那么这封邮件首先通过 SMTP 协议发送到本地邮件服务器 mail.domain1.edu.cn，本地邮件服务器 mail.domain1.edu.cn 根据收到的邮件中的目标邮件地址 lisi@domain2.edu.cn 向 DNS 查询域 domain2.edu.cn 中 MX 记录来获得该域中的邮件服务器地址，然后，mail.domain1.edu.cn 和 mail.domain2.edu.cn 之间建立 TCP 连接，通过 SMTP 协议传输，这样，就完成了邮件从发送方到接收方的传递过程，邮件存储在 mail.domain2.edu.cn 中等待用户接受。

从上面的邮件传递过程可以看到，SMTP 工作有两种情况：一是电子邮件从客户机传输到服务器；二是从某一个服务器传输到另一个服务器。文件 RFC821 规定了该协议的所有细节。

SMTP 是个请求/响应协议，命令和响应用 NVT ASCII 字符，并以 CR 和 LF 符结束。响应包括一个表示返回状态的三位数字代码，SMTP 在 TCP 协议 25 号端口监听连接请求。

图 9-27 为 SMTP 的通信模型示意图。

从图 9-27 可以看到 SMTP 协议在发送 SMTP 和接收 SMTP 之间的会话是靠发送 SMTP 的 SMTP 命令和接收 SMTP 反馈的应答来完成的。SMTP 协议对每一个命令都会返回一个应答码，应答码的每一个数字都是有特定含义的，如第一位数字为 2 时表示命令成

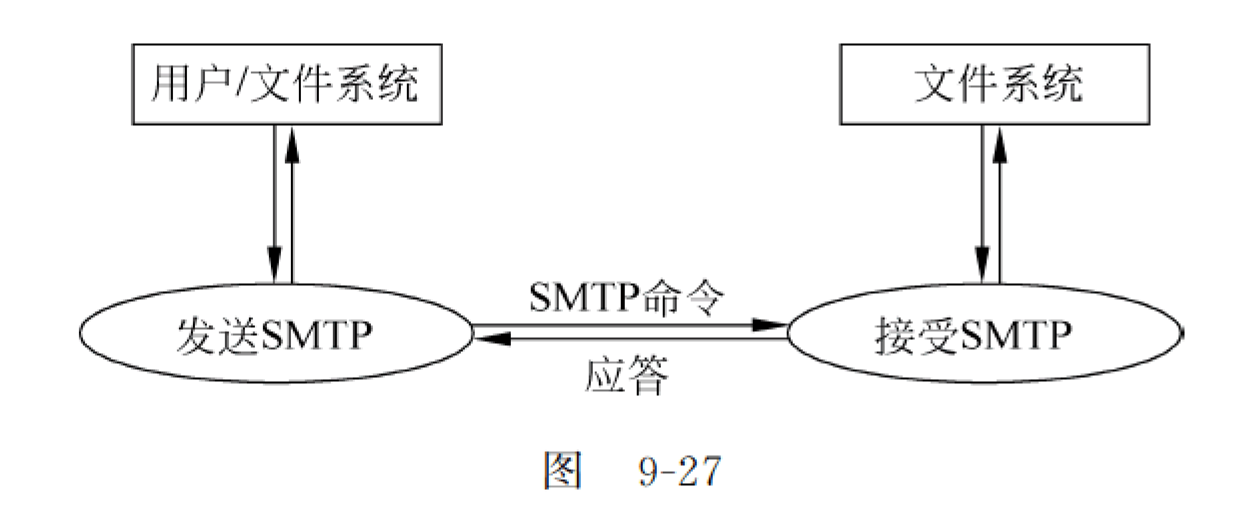



图　9-27

功,为 5 表失败。一些较复杂的邮件程序利用该特点,首先检查应答码的首数字,并根据其值来决定下一步的动作。

SMTP 的应答码序列如下:

211　系统状态或系统帮助响应
214　帮助信息
220　＜domain＞服务就绪
221　＜domain＞服务关闭
250　要求的邮件操作完成
251　用户非本地,将转发向＜forward-path＞
354　开始邮件输入,以“.”结束
421　＜domain＞服务未就绪,关闭传输信道
450　要求的邮件操作未完成,邮箱不可用
451　放弃要求的操作;处理过程中出错
501　参数格式错误
502　命令不可实现
503　错误的命令序列
504　命令参数不可实现
550　要求的邮件操作未完成,邮箱不可用
551　用户非本地,请尝试＜forward-path＞
553　邮箱名不可用,要求的操作未执行
554　操作失败

SMTP 基本命令集如表 9-2 所示。

**表 9-2　SMTP 基本命令集**

| 命令 | 描　　述 |
|---|---|
| HELO | 发件方问候收件方,后面跟的是发件人的服务器地址或标识。收件方回答 OK 时标识自己的身份。问候和确认过程表明两台机器可以进行通信,同时状态参量被复位,缓冲区被清空 |
| MAIL | 用来开始传送邮件,它的后面跟随发件方邮件地址(返回邮件地址)。它也用来当邮件无法送达时,发送失败通知。为保证邮件的成功发送,发件方的地址应是被对方或中间转发方同意接受的。这个命令会清空有关的缓冲区,为新的邮件做准备<br>mail　from: |

续表

| 命令 | 描　　述 |
|---|---|
| RCPT | 这个命令告诉收件方收件人的邮箱。当有多个收件人时，需要多次使用该命令，每次只能指明一个人。如果接收方服务器不同意转发这个地址的邮件，它必须报550错误代码通知发件方。如果服务器同意转发，它要更改邮件发送路径，把最开始的目的地(该服务器)换成下一个服务器。 |
| DATA | 收件方把该命令之后的数据作为发送的数据。数据被加入数据缓冲区中，以单独一行是<CRLF>.<CRLF>的行结束数据。结束行对于接收方同时意味着立即开始缓冲区内的数据传送，传送结束后清空缓冲区。如果传送接受，接收方回复OK |
| VRFY | 用于验证指定的用户/邮箱是否存在 |
| EXPN | 验证给定的邮箱列表是否存在，扩充邮箱列表，常被禁用 |
| HELP | 查询服务器支持什么命令 |
| NOOP | 这个命令不影响任何参数，只是要求接收放回答OK，不会影响缓冲区的数据 |
| QUIT | SMTP要求接收方必须回答OK，然后中断传输；在收到这个命令并回答OK前，收件方不得中断连接，即使传输出现错误。发件方在发出这个命令并收到OK答复前，也不得中断连接 |
| RSET | 这个命令用来通知收件方复位，所有已存入缓冲区的收件人数据，发件人数据和待传送的数据都必须清除，接收放必须回答OK |

一封邮件的发送要经过好几次传递，建立好几次SMTP会话，接下来看SMTP会话是如何建立的：

(1) 建立TCP连接。

(2) 客户端发送EHLO命令以标识发件人自己的身份，然后客户端发送MAIL命令，服务器端以OK作为响应，表明准备接收在EHLO命令之后，接着电子邮件程序会发送MAIL命令。MAIL命令标识出发送者，它有两个参数“FROM:”和一个电子邮件地址。如果SMTP服务程序能够成功地解析电子邮件地址的话，通常它将返回以250开头的回应消息；否则将发送回表示操作失败的回应消息。

(1) 客户端发送RCPT命令，以标识该电子邮件的计划接收人，可以有多个RCPT行，服务器端发回响应表示是否愿意为收件人接受邮件。

(2) 协商结束，发送邮件，用命令DATA发送。

(3) 以“.”表示结束输入内容并一起发送出去。

(4) 结束此次发送，用QUIT命令退出。

下面看一个SMTP的实例。

客户端和服务器端通过TCP三次握手建立TCP连接，如图9-28所示服务器端口为25。

服务器端发送响应，应答码为220，表示服务器准备就绪，如图9-29所示。

客户端发送EHLO命令，表示开始SMTP会话，如图9-30所示。

**注意**：EHLO命令是扩展SMTP命令集中的一个命令，支持SMTP服务扩展的客户应该以EHLO命令开始SMTP会话，而不是通常的HELO命令。如果服务器也支持，那就返回确认响应，如果不支持就返回失败响应。因为引入了EHLO命令，因此会话开始的第一条命令可以是HELO或EHLO。

服务器端返回SMTP响应，应答码为250，表示请求建立的邮件服务会话已经就绪，如图9-31所示。

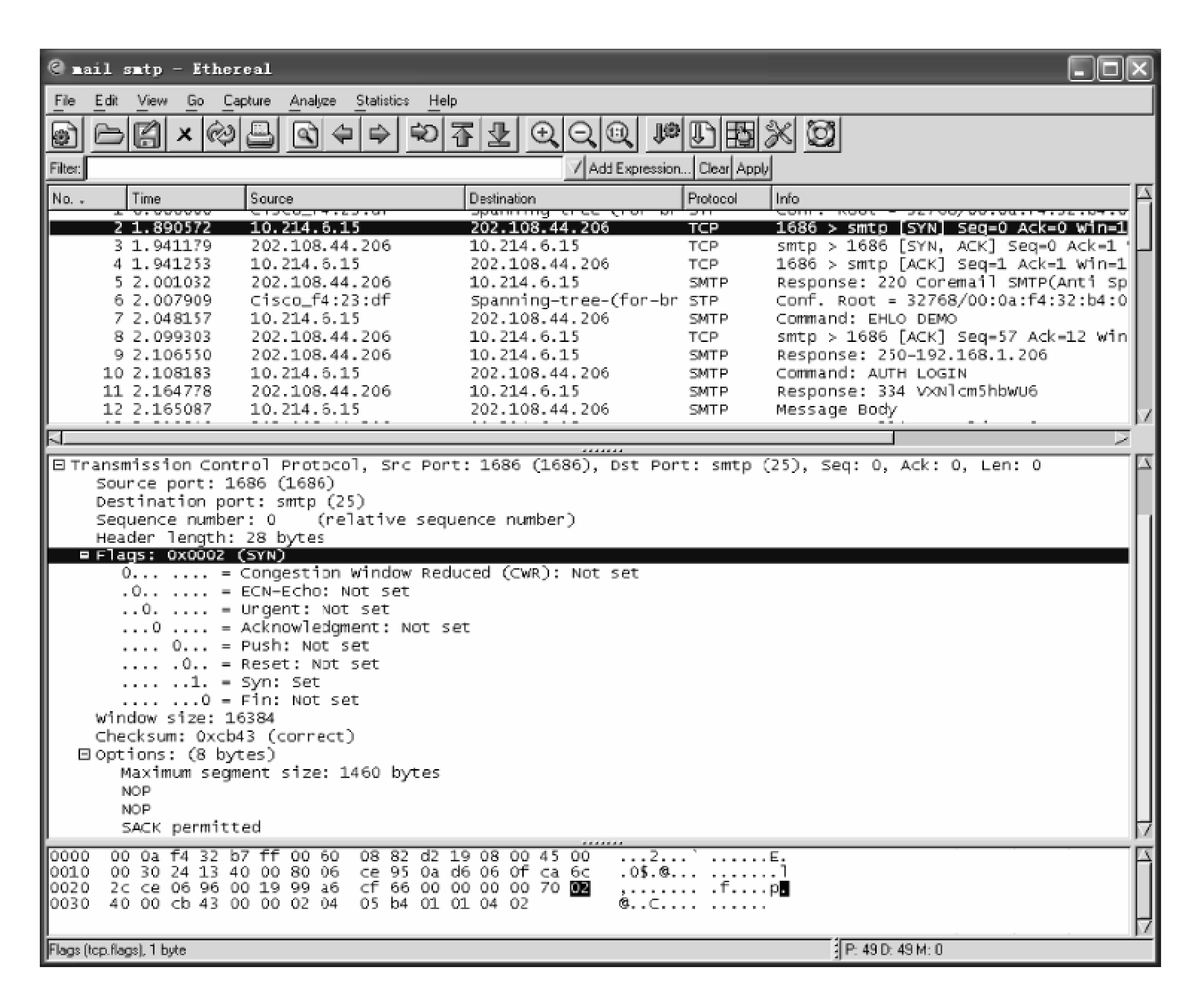

mail smtp - Ethereal
File Edit View Go Capture Analyze Statistics Help
Filter: Add Expression... Clear Apply
No. . Time Source Destination Protocol Info
2 1.890572 10.214.6.15 202.108.44.206 TCP 1686 > smtp [SYN] Seq=0 Ack=0 Win=1
3 1.941179 202.108.44.206 10.214.6.15 TCP smtp > 1686 [SYN, ACK] Seq=0 Ack=1
4 1.941253 10.214.6.15 202.108.44.206 TCP 1686 > smtp [ACK] Seq=1 Ack=1 Win=1
5 2.001032 202.108.44.206 10.214.6.15 SMTP Response: 220 Coremail SMTP(Anti Sp
6 2.007909 Cisco_f4:23:df Spanning-tree-(for-br STP Conf. Root = 32768/00:0a:f4:32:b4:0
7 2.048157 10.214.6.15 202.108.44.206 SMTP Command: EHLO DEMO
8 2.099303 202.108.44.206 10.214.6.15 TCP smtp > 1686 [ACK] Seq=57 Ack=12 Win
9 2.106550 202.108.44.206 10.214.6.15 SMTP Response: 250-192.168.1.206
10 2.108183 10.214.6.15 202.108.44.206 SMTP Command: AUTH LOGIN
11 2.164778 202.108.44.206 10.214.6.15 SMTP Response: 334 VXNlcm5hbWU6
12 2.165087 10.214.6.15 202.108.44.206 SMTP Message Body
Transmission Control Protocol, Src Port: 1686 (1686), Dst Port: smtp (25), Seq: 0, Ack: 0, Len: 0
Source port: 1686 (1686)
Destination port: smtp (25)
Sequence number: 0 (relative sequence number)
Header length: 28 bytes
Flags: 0x0002 (SYN)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...0 .... = Acknowledgment: Not set
.... 0... = Push: Not set
.... .0.. = Reset: Not set
.... ..1. = Syn: Set
.... ...0 = Fin: Not set
Window size: 16384
Checksum: 0xcb43 (correct)
Options: (8 bytes)
Maximum segment size: 1460 bytes
NOP
NOP
SACK permitted
0000 00 0a f4 32 b7 ff 00 60 08 82 d2 19 08 00 45 00
0010 00 30 24 13 40 00 80 06 ce 95 0a d6 06 0f ca 6c
0020 2c ce 06 96 00 19 99 a6 cf 66 00 00 00 00 70 02
0030 40 00 cb 43 00 00 02 04 05 b4 01 01 04 02
Flags (tcp.flags), 1 byte
P: 49 D: 49 M: 0

图　9-28

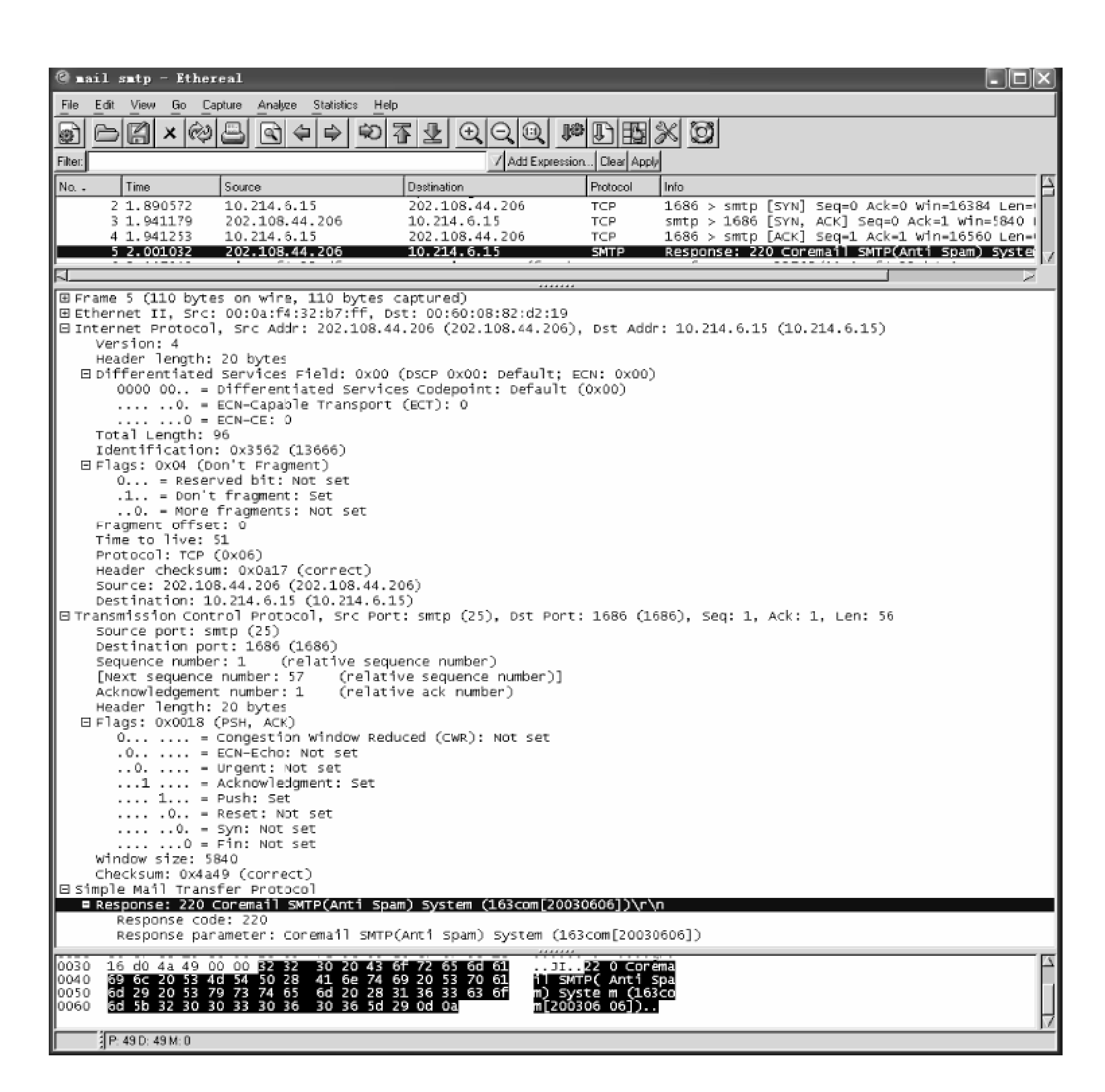

mail smtp - Ethereal
File Edit View Go Capture Analyze Statistics Help
Filter: Add Expression... Clear Apply
No. . Time Source Destination Protocol Info
2 1.890572 10.214.6.15 202.108.44.206 TCP 1686 > smtp [SYN] Seq=0 Ack=0 Win=16384 Len=
3 1.941179 202.108.44.206 10.214.6.15 TCP smtp > 1686 [SYN, ACK] Seq=0 Ack=1 Win=5840 L
4 1.941253 10.214.6.15 202.108.44.206 TCP 1686 > smtp [ACK] Seq=1 Ack=1 Win=16560 Len=
5 2.001032 202.108.44.206 10.214.6.15 SMTP Response: 220 Coremail SMTP(Anti Spam) Syste
Frame 5 (110 bytes on wire, 110 bytes captured)
Ethernet II, Src: 00:0a:f4:32:b7:ff, Dst: 00:60:08:82:d2:19
Internet Protocol, Src Addr: 202.108.44.206 (202.108.44.206), Dst Addr: 10.214.6.15 (10.214.6.15)
Version: 4
Header length: 20 bytes
Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
0000 00.. = Differentiated Services Codepoint: Default (0x00)
.... ..0. = ECN-Capable Transport (ECT): 0
.... ...0 = ECN-CE: 0
Total Length: 96
Identification: 0x3562 (13666)
Flags: 0x04 (Don't Fragment)
0... = Reserved bit: Not set
.1.. = Don't fragment: Set
..0. = More fragments: Not set
Fragment offset: 0
Time to live: 51
Protocol: TCP (0x06)
Header checksum: 0x0a17 (correct)
Source: 202.108.44.206 (202.108.44.206)
Destination: 10.214.6.15 (10.214.6.15)
Transmission Control Protocol, Src Port: smtp (25), Dst Port: 1686 (1686), Seq: 1, Ack: 1, Len: 56
Source port: smtp (25)
Destination port: 1686 (1686)
Sequence number: 1 (relative sequence number)
[Next sequence number: 57 (relative sequence number)]
Acknowledgement number: 1 (relative ack number)
Header length: 20 bytes
Flags: 0x0018 (PSH, ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 1... = Push: Set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 5840
Checksum: 0x4a49 (correct)
Simple Mail Transfer Protocol
Response: 220 Coremail SMTP(Anti Spam) System (163com[20030606])\r\n
Response code: 220
Response parameter: Coremail SMTP(Anti Spam) System (163com[20030606])
0030 16 d0 4a 49 00 00 32 32 30 20 43 6f 72 65 6d 61
0040 69 6c 20 53 4d 54 50 28 41 6e 74 69 20 53 70 61
0050 6d 29 20 53 79 73 74 65 6d 20 28 31 36 33 63 6f
0060 6d 5b 32 30 30 33 30 36 30 36 5d 29 0d 0a
P: 49 D: 49 M: 0

图　9-29

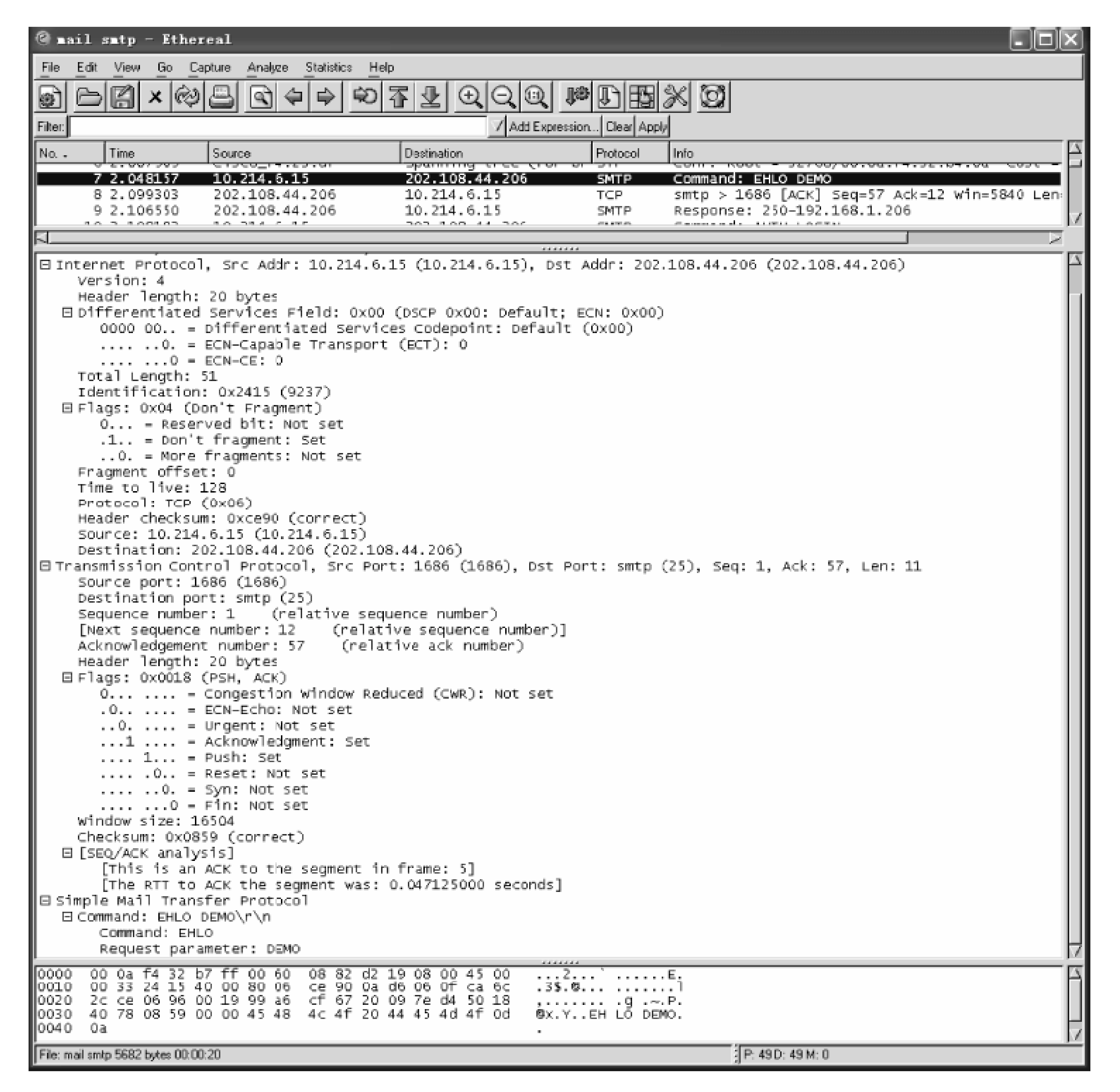


图 9-30

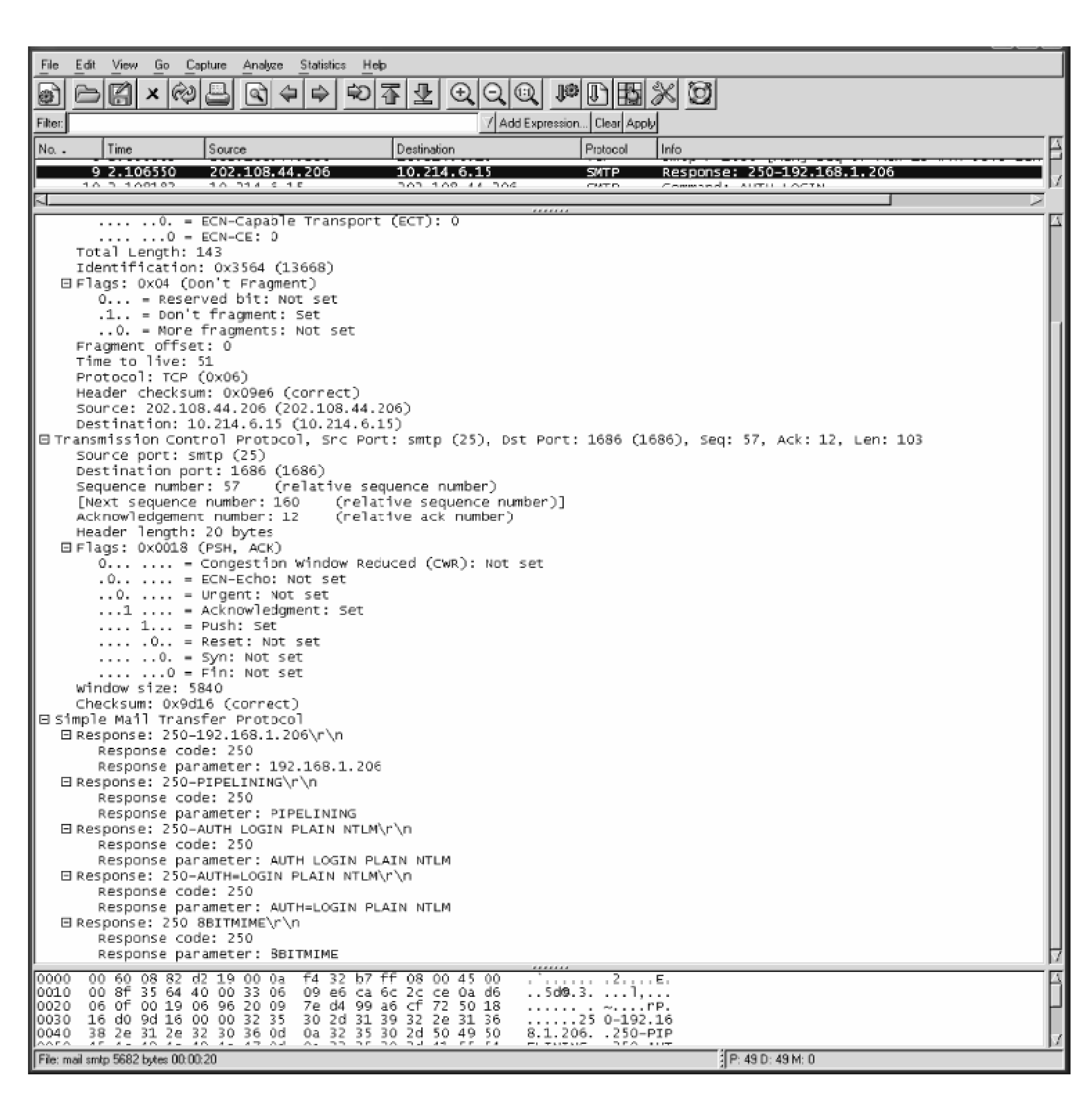


图 9-31

出于安全的考虑，SMTP 服务器要求发送邮件时，对发送者进行身份认证，客户端发送 AUTH LOGIN 命令，如图 9-32 所示。




图　9-32

服务器返回应答，应答码为 334，还可以看到一些用 base64 编码的字符串文本，该字符串的意思为 username，如图 9-33 所示。

**注意：**对于 LOGIN 方式认证，其实就是将用户名与口令用 base64 进行编码，根据服务器的要求，分别发出即可。

接下来客户端发送用 base64 编码的用户名给服务器，如图 9-34 所示。

服务器发回响应，响应码为 334，同时返回 base64 编码串，意思为 password，如图 9-35 所示。

客户端发送用 base64 编码的密码给服务器，如图 9-36 所示。

服务器端发送 TCP 确认，如图 9-37 所示。

服务端返回码为 235，表示认证成功可以发送邮件了，如图 9-38 所示。

客户端发送"MAIL FORM："命令用来告诉服务器发送者的邮件地址，如图 9-39 所示。

服务器返回一个 TCP 确认，如图 9-40 所示。

服务器返回 SMTP 响应，应答码为 250，表示操作成功，服务器就绪，如图 9-41 所示。

客户端用"RCPT TO："命令来指定邮件接收者的邮件地址，如图 9-42 所示。

服务器返回 SMTP 响应，应答码为 250，表示操作成功，服务器就绪，如图 9-43 所示。

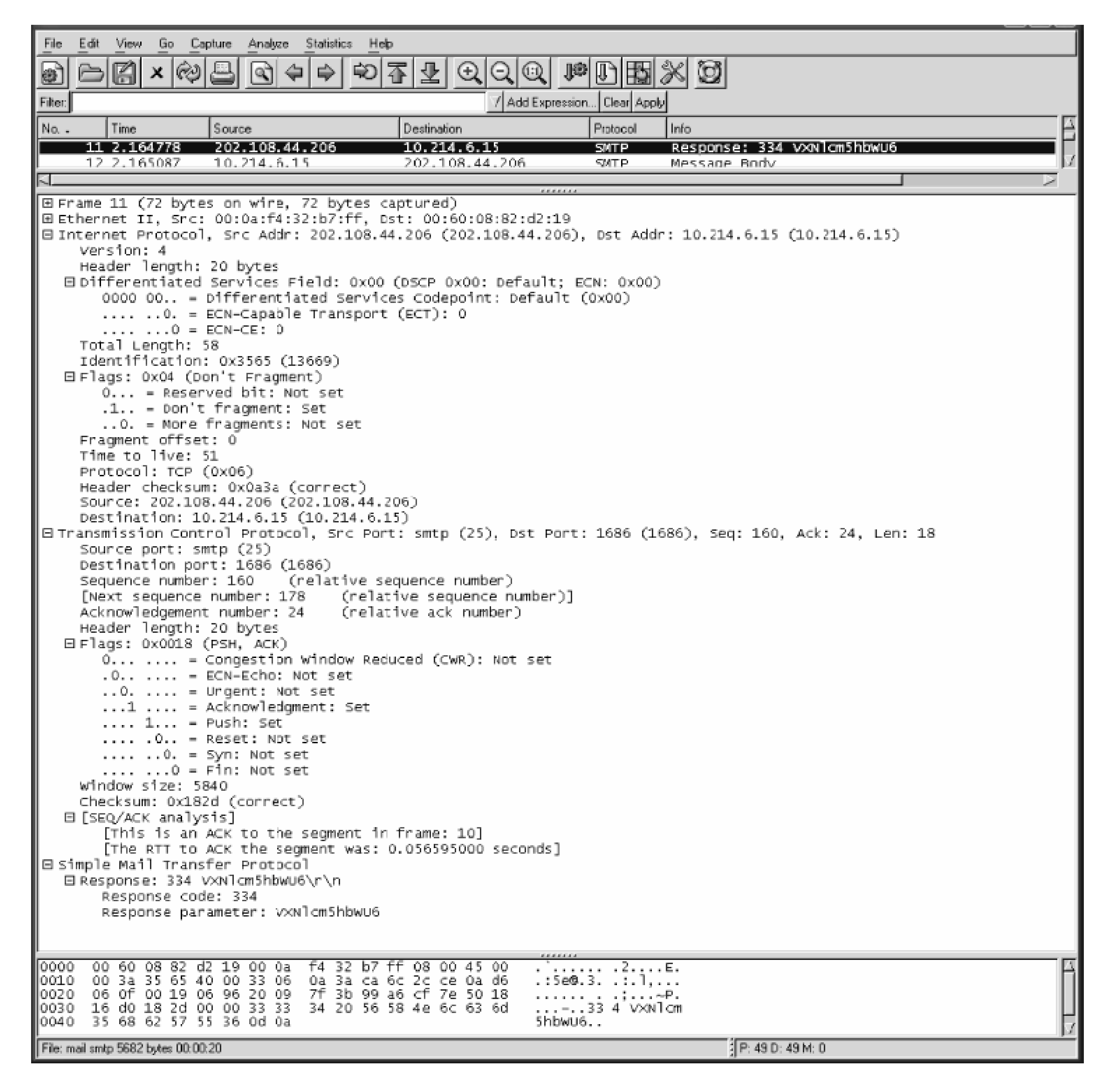

File Edit View Go Capture Analyze Statistics Help
Filter:
Add Expression... Clear Apply
No. Time Source Destination Protocol Info
11 2.164778 202.108.44.206 10.214.6.15 SMTP Response: 334 VXNlcm5hbWU6
12 2.165087 10.214.6.15 202.108.44.206 SMTP Message Body
Frame 11 (72 bytes on wire, 72 bytes captured)
Ethernet II, Src: 00:0a:f4:32:b7:ff, Dst: 00:60:08:82:d2:19
Internet Protocol, Src Addr: 202.108.44.206 (202.108.44.206), Dst Addr: 10.214.6.15 (10.214.6.15)
Version: 4
Header length: 20 bytes
Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
0000 00.. = Differentiated Services Codepoint: Default (0x00)
.... ..0. = ECN-Capable Transport (ECT): 0
.... ...0 = ECN-CE: 0
Total Length: 58
Identification: 0x3565 (13669)
Flags: 0x04 (Don't Fragment)
0... = Reserved bit: Not set
.1.. = Don't fragment: Set
..0. = More fragments: Not set
Fragment offset: 0
Time to live: 51
Protocol: TCP (0x06)
Header checksum: 0x0a3a (correct)
Source: 202.108.44.206 (202.108.44.206)
Destination: 10.214.6.15 (10.214.6.15)
Transmission Control Protocol, Src Port: smtp (25), Dst Port: 1686 (1686), Seq: 160, Ack: 24, Len: 18
Source port: smtp (25)
Destination port: 1686 (1686)
Sequence number: 160 (relative sequence number)
[Next sequence number: 178 (relative sequence number)]
Acknowledgement number: 24 (relative ack number)
Header length: 20 bytes
Flags: 0x0018 (PSH, ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 1... = Push: Set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 5840
Checksum: 0x182d (correct)
[SEQ/ACK analysis]
[This is an ACK to the segment in frame: 10]
[The RTT to ACK the segment was: 0.056595000 seconds]
Simple Mail Transfer Protocol
Response: 334 VXNlcm5hbWU6\r\n
Response code: 334
Response parameter: VXNlcm5hbWU6
0000 00 60 08 82 d2 19 00 0a f4 32 b7 ff 08 00 45 00
0010 00 3a 35 65 40 00 33 06 0a 3a ca 6c 2c ce 0a d6
0020 06 0f 00 19 06 96 20 09 7f 3b 99 a6 cf 7e 50 18
0030 16 d0 18 2d 00 00 33 33 34 20 56 58 4e 6c 63 6d
0040 35 68 62 57 55 36 0d 0a
File: mail smtp 5682 bytes 00:00:20
P: 49 D: 49 M: 0

图 9-33

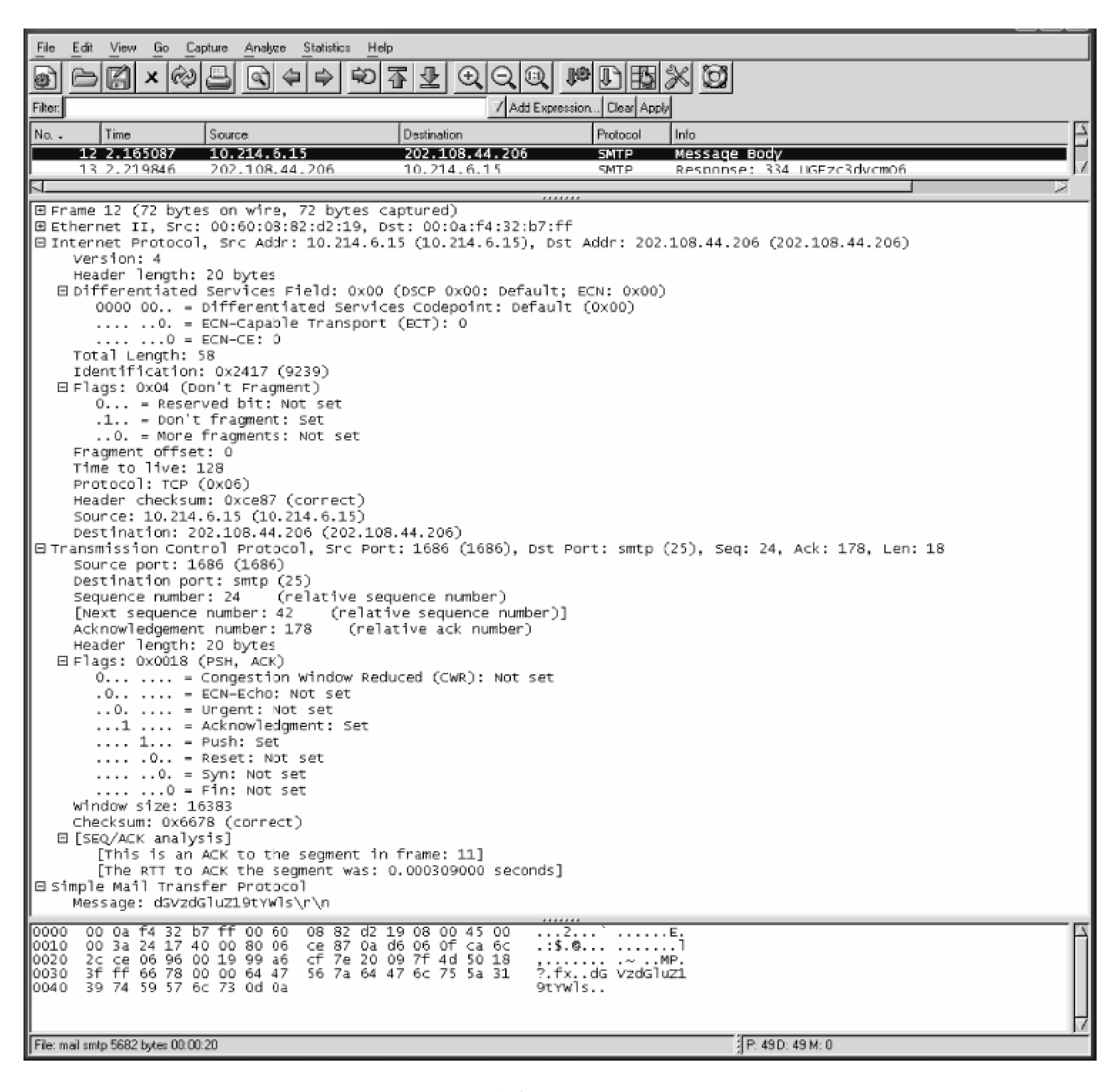

File Edit View Go Capture Analyze Statistics Help
Filter:
Add Expression... Clear Apply
No. Time Source Destination Protocol Info
12 2.165087 10.214.6.15 202.108.44.206 SMTP Message Body
13 2.219846 202.108.44.206 10.214.6.15 SMTP Response: 334 UGFzc3dvcm06
Frame 12 (72 bytes on wire, 72 bytes captured)
Ethernet II, Src: 00:60:08:82:d2:19, Dst: 00:0a:f4:32:b7:ff
Internet Protocol, Src Addr: 10.214.6.15 (10.214.6.15), Dst Addr: 202.108.44.206 (202.108.44.206)
Version: 4
Header length: 20 bytes
Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
0000 00.. = Differentiated Services Codepoint: Default (0x00)
.... ..0. = ECN-Capable Transport (ECT): 0
.... ...0 = ECN-CE: 0
Total Length: 58
Identification: 0x2417 (9239)
Flags: 0x04 (Don't Fragment)
0... = Reserved bit: Not set
.1.. = Don't fragment: Set
..0. = More fragments: Not set
Fragment offset: 0
Time to live: 128
Protocol: TCP (0x06)
Header checksum: 0xce87 (correct)
Source: 10.214.6.15 (10.214.6.15)
Destination: 202.108.44.206 (202.108.44.206)
Transmission Control Protocol, Src Port: 1686 (1686), Dst Port: smtp (25), Seq: 24, Ack: 178, Len: 18
Source port: 1686 (1686)
Destination port: smtp (25)
Sequence number: 24 (relative sequence number)
[Next sequence number: 42 (relative sequence number)]
Acknowledgement number: 178 (relative ack number)
Header length: 20 bytes
Flags: 0x0018 (PSH, ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 1... = Push: Set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 16383
Checksum: 0x6678 (correct)
[SEQ/ACK analysis]
[This is an ACK to the segment in frame: 11]
[The RTT to ACK the segment was: 0.000309000 seconds]
Simple Mail Transfer Protocol
Message: dGVzdGluZzl9tYWls\r\n
0000 00 0a f4 32 b7 ff 00 60 08 82 d2 19 08 00 45 00
0010 00 3a 24 17 40 00 80 06 ce 87 0a d6 06 0f ca 6c
0020 2c ce 06 96 00 19 99 a6 cf 7e 20 09 7f 4d 50 18
0030 3f ff 66 78 00 00 64 47 56 7a 64 47 6c 75 5a 31
0040 39 74 59 57 6c 73 0d 0a
File: mail smtp 5682 bytes 00:00:20
P: 49 D: 49 M: 0

图 9-34

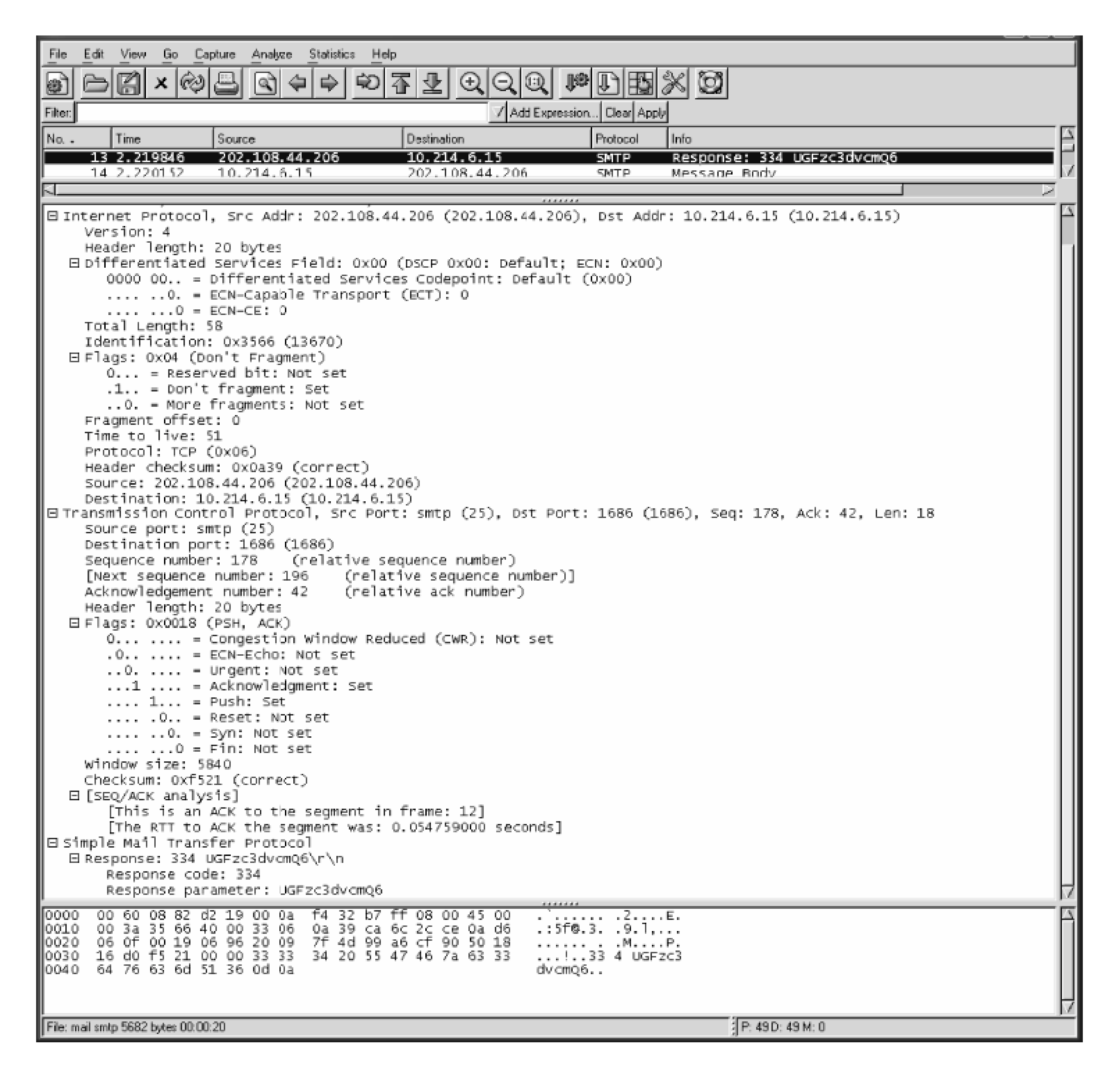

File Edit View Go Capture Analyze Statistics Help
Filter: Add Expression... Clear Apply
No. Time Source Destination Protocol Info
13 2.219846 202.108.44.206 10.214.6.15 SMTP Response: 334 UGFzc3dvcmQ6
14 2.220132 10.214.6.15 202.108.44.206 SMTP Message Body
Internet Protocol, Src Addr: 202.108.44.206 (202.108.44.206), Dst Addr: 10.214.6.15 (10.214.6.15)
Version: 4
Header length: 20 bytes
Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
0000 00.. = Differentiated Services Codepoint: Default (0x00)
.... ..0. = ECN-Capable Transport (ECT): 0
.... ...0 = ECN-CE: 0
Total Length: 58
Identification: 0x3566 (13670)
Flags: 0x04 (Don't Fragment)
0... = Reserved bit: Not set
.1.. = Don't fragment: Set
..0. = More fragments: Not set
Fragment offset: 0
Time to live: 51
Protocol: TCP (0x06)
Header checksum: 0x0a39 (correct)
Source: 202.108.44.206 (202.108.44.206)
Destination: 10.214.6.15 (10.214.6.15)
Transmission Control Protocol, Src Port: smtp (25), Dst Port: 1686 (1686), Seq: 178, Ack: 42, Len: 18
Source port: smtp (25)
Destination port: 1686 (1686)
Sequence number: 178 (relative sequence number)
[Next sequence number: 196 (relative sequence number)]
Acknowledgement number: 42 (relative ack number)
Header length: 20 bytes
Flags: 0x0018 (PSH, ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 1... = Push: Set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 5840
Checksum: 0xf521 (correct)
[SEQ/ACK analysis]
[This is an ACK to the segment in frame: 12]
[The RTT to ACK the segment was: 0.054759000 seconds]
Simple Mail Transfer Protocol
Response: 334 UGFzc3dvcmQ6\r\n
Response code: 334
Response parameter: UGFzc3dvcmQ6
0000 00 60 08 82 d2 19 00 0a f4 32 b7 ff 08 00 45 00
0010 00 3a 35 66 40 00 33 06 0a 39 ca 6c 2c ce 0a d6
0020 06 0f 00 19 06 96 20 09 7f 4d 99 a6 cf 90 50 18
0030 16 d0 f5 21 00 00 33 33 34 20 55 47 46 7a 63 33
0040 64 76 63 6d 51 36 0d 0a
File: mail smtp 5682 bytes 00:00:20
P: 49 D: 49 M: 0

图　9-35

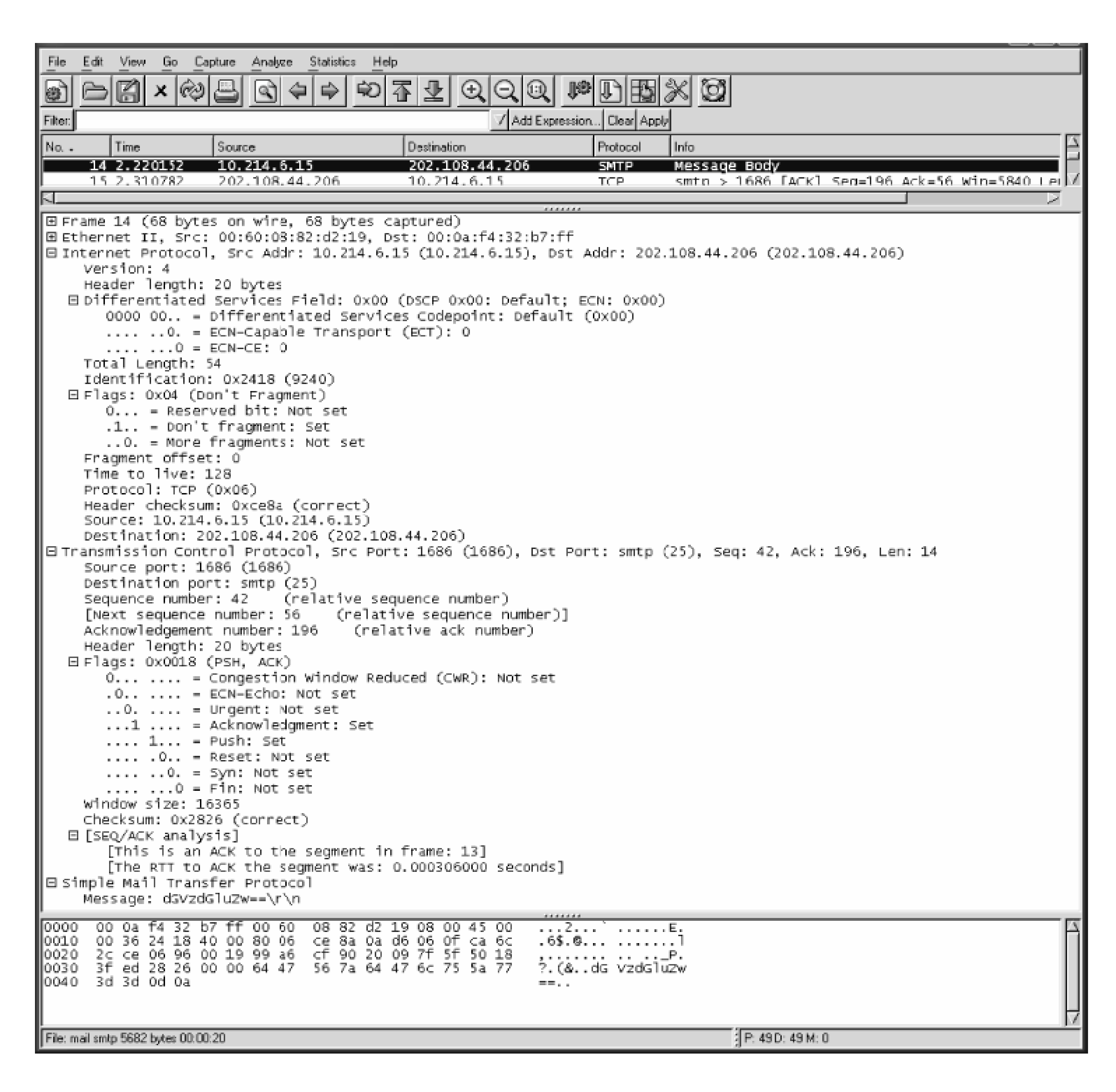

File Edit View Go Capture Analyze Statistics Help
Filter: Add Expression... Clear Apply
No. Time Source Destination Protocol Info
14 2.220132 10.214.6.15 202.108.44.206 SMTP Message Body
15 2.310782 202.108.44.206 10.214.6.15 TCP smtp > 1686 [ACK] Seq=196 Ack=56 Win=5840
Frame 14 (68 bytes on wire, 68 bytes captured)
Ethernet II, Src: 00:60:08:82:d2:19, Dst: 00:0a:f4:32:b7:ff
Internet Protocol, Src Addr: 10.214.6.15 (10.214.6.15), Dst Addr: 202.108.44.206 (202.108.44.206)
Version: 4
Header length: 20 bytes
Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
0000 00.. = Differentiated Services Codepoint: Default (0x00)
.... ..0. = ECN-Capable Transport (ECT): 0
.... ...0 = ECN-CE: 0
Total Length: 54
Identification: 0x2418 (9240)
Flags: 0x04 (Don't Fragment)
0... = Reserved bit: Not set
.1.. = Don't fragment: Set
..0. = More fragments: Not set
Fragment offset: 0
Time to live: 128
Protocol: TCP (0x06)
Header checksum: 0xce8a (correct)
Source: 10.214.6.15 (10.214.6.15)
Destination: 202.108.44.206 (202.108.44.206)
Transmission Control Protocol, Src Port: 1686 (1686), Dst Port: smtp (25), Seq: 42, Ack: 196, Len: 14
Source port: 1686 (1686)
Destination port: smtp (25)
Sequence number: 42 (relative sequence number)
[Next sequence number: 56 (relative sequence number)]
Acknowledgement number: 196 (relative ack number)
Header length: 20 bytes
Flags: 0x0018 (PSH, ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 1... = Push: Set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 16365
Checksum: 0x2826 (correct)
[SEQ/ACK analysis]
[This is an ACK to the segment in frame: 13]
[The RTT to ACK the segment was: 0.000306000 seconds]
Simple Mail Transfer Protocol
Message: dSVzdGluZw==\r\n
0000 00 0a f4 32 b7 ff 00 60 08 82 d2 19 08 00 45 00
0010 00 36 24 18 40 00 80 06 ce 8a 0a d6 06 0f ca 6c
0020 2c ce 06 96 00 19 99 a6 cf 90 20 09 7f 5f 50 18
0030 3f ed 28 26 00 00 64 47 56 7a 64 47 6c 75 5a 77
0040 3d 3d 0d 0a
File: mail smtp 5682 bytes 00:00:20
P: 49 D: 49 M: 0

图　9-36

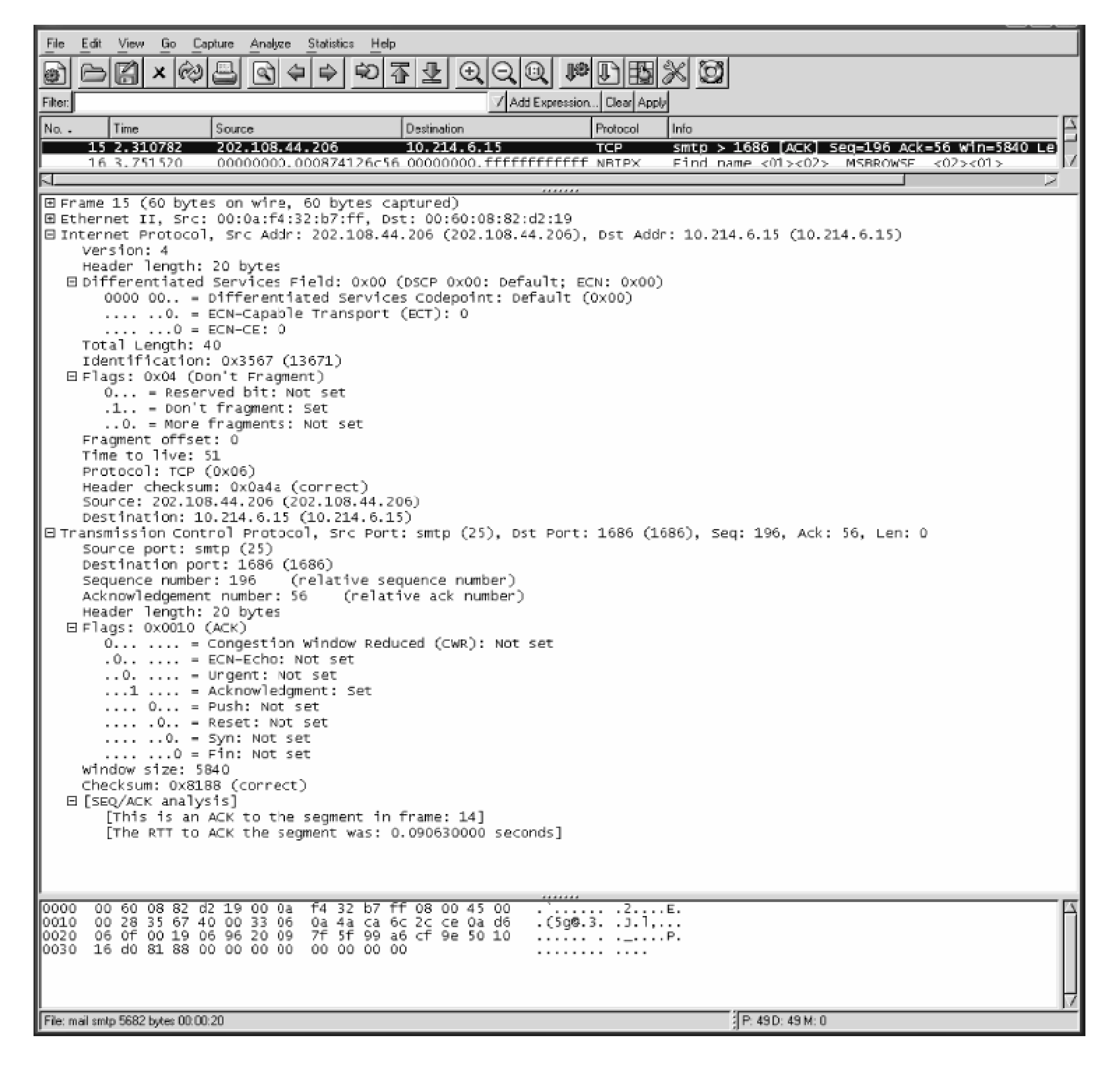

File Edit View Go Capture Analyze Statistics Help
Filter:
Add Expression... Clear Apply
No. Time Source Destination Protocol Info
15 2.310782 202.108.44.206 10.214.6.15 TCP smtp > 1686 [ACK] Seq=196 Ack=56 Win=5840 Le
16 3.751520 00000000.000874126c56 00000000.ffffffffffff NBIPX Find name <01><02> MSBROWSE <02><01>
Frame 15 (60 bytes on wire, 60 bytes captured)
Ethernet II, Src: 00:0a:f4:32:b7:ff, Dst: 00:60:08:82:d2:19
Internet Protocol, Src Addr: 202.108.44.206 (202.108.44.206), Dst Addr: 10.214.6.15 (10.214.6.15)
Version: 4
Header length: 20 bytes
Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
0000 00.. = Differentiated Services Codepoint: Default (0x00)
.... ..0. = ECN-Capable Transport (ECT): 0
.... ...0 = ECN-CE: 0
Total Length: 40
Identification: 0x3567 (13671)
Flags: 0x04 (Don't Fragment)
0... = Reserved bit: Not set
.1.. = Don't fragment: Set
..0. = More fragments: Not set
Fragment offset: 0
Time to live: 51
Protocol: TCP (0x06)
Header checksum: 0x0a4a (correct)
Source: 202.108.44.206 (202.108.44.206)
Destination: 10.214.6.15 (10.214.6.15)
Transmission Control Protocol, Src Port: smtp (25), Dst Port: 1686 (1686), Seq: 196, Ack: 56, Len: 0
Source port: smtp (25)
Destination port: 1686 (1686)
Sequence number: 196 (relative sequence number)
Acknowledgement number: 56 (relative ack number)
Header length: 20 bytes
Flags: 0x0010 (ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 0... = Push: Not set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 5840
Checksum: 0x8188 (correct)
[SEQ/ACK analysis]
[This is an ACK to the segment in frame: 14]
[The RTT to ACK the segment was: 0.090630000 seconds]
0000 00 60 08 82 d2 19 00 0a f4 32 b7 ff 08 00 45 00
0010 00 28 35 67 40 00 33 06 0a 4a ca 6c 2c ce 0a d6
0020 06 0f 00 19 06 96 20 09 7f 5f 99 a6 cf 9e 50 10
0030 16 d0 81 88 00 00 00 00 00 00 00 00
File: mail smtp 5682 bytes 00:00:20
P: 49 D: 49 M: 0

图 9-37

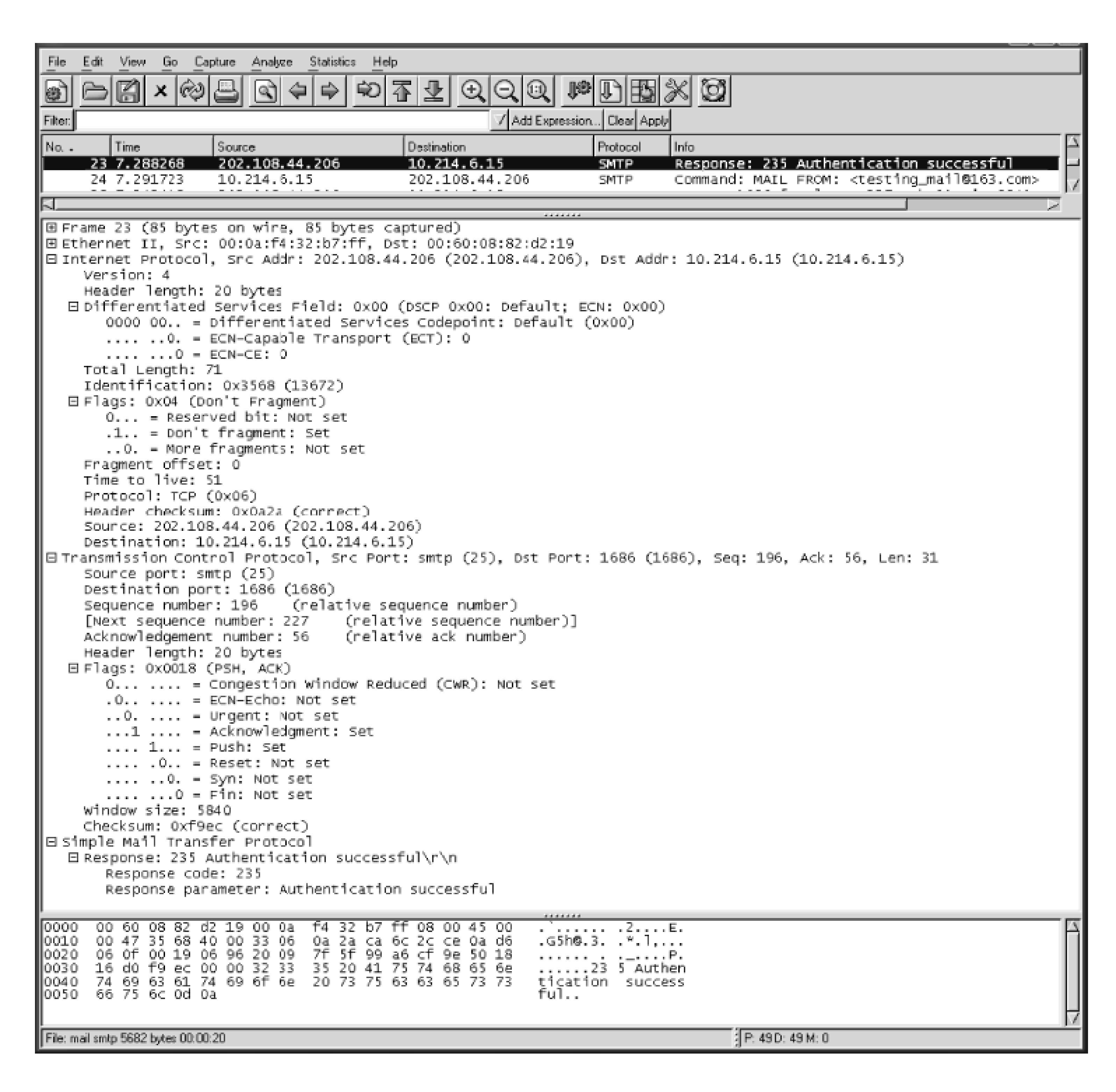

File Edit View Go Capture Analyze Statistics Help
Filter:
Add Expression... Clear Apply
No. Time Source Destination Protocol Info
23 7.288268 202.108.44.206 10.214.6.15 SMTP Response: 235 Authentication successful
24 7.291723 10.214.6.15 202.108.44.206 SMTP Command: MAIL FROM: <testing_mail@163.com>
Frame 23 (85 bytes on wire, 85 bytes captured)
Ethernet II, Src: 00:0a:f4:32:b7:ff, Dst: 00:60:08:82:d2:19
Internet Protocol, Src Addr: 202.108.44.206 (202.108.44.206), Dst Addr: 10.214.6.15 (10.214.6.15)
Version: 4
Header length: 20 bytes
Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
0000 00.. = Differentiated Services Codepoint: Default (0x00)
.... ..0. = ECN-Capable Transport (ECT): 0
.... ...0 = ECN-CE: 0
Total Length: 71
Identification: 0x3568 (13672)
Flags: 0x04 (Don't Fragment)
0... = Reserved bit: Not set
.1.. = Don't fragment: Set
..0. = More fragments: Not set
Fragment offset: 0
Time to live: 51
Protocol: TCP (0x06)
Header checksum: 0x0a2a (correct)
Source: 202.108.44.206 (202.108.44.206)
Destination: 10.214.6.15 (10.214.6.15)
Transmission Control Protocol, Src Port: smtp (25), Dst Port: 1686 (1686), Seq: 196, Ack: 56, Len: 31
Source port: smtp (25)
Destination port: 1686 (1686)
Sequence number: 196 (relative sequence number)
[Next sequence number: 227 (relative sequence number)]
Acknowledgement number: 56 (relative ack number)
Header length: 20 bytes
Flags: 0x0018 (PSH, ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 1... = Push: Set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 5840
Checksum: 0xf9ec (correct)
Simple Mail Transfer Protocol
Response: 235 Authentication successful\r\n
Response code: 235
Response parameter: Authentication successful
0000 00 60 08 82 d2 19 00 0a f4 32 b7 ff 08 00 45 00
0010 00 47 35 68 40 00 33 06 0a 2a ca 6c 2c ce 0a d6
0020 06 0f 00 19 06 96 20 09 7f 5f 99 a6 cf 9e 50 18
0030 16 d0 f9 ec 00 00 32 33 35 20 41 75 74 68 65 6e
0040 74 69 63 61 74 69 6f 6e 20 73 75 63 63 65 73 73
0050 66 75 6c 0d 0a
File: mail smtp 5682 bytes 00:00:20
P: 49 D: 49 M: 0

图 9-38

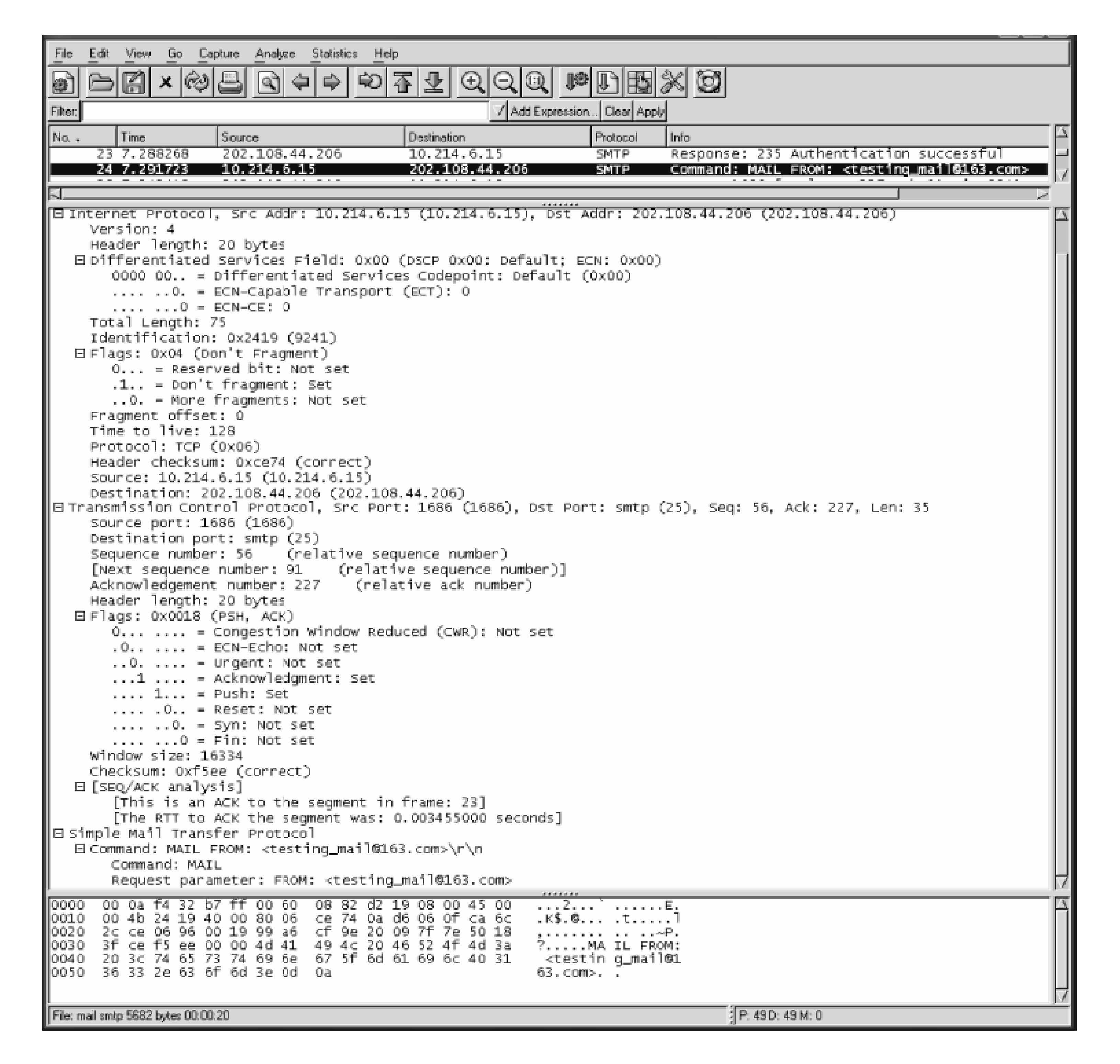



图　9-39

```
File Edit View Go Capture Analyze Statistics Help
Filter:                                   Add Expression... Clear Apply
No. .  Time       Source           Destination     Protocol  Info
 25 7.342415  202.108.44.206   10.214.6.15     TCP       smtp > 1686 [ACK] Seq=227 Ack=91 Win=5840 Le
 26 7.349027  202.108.44.206   10.214.6.15     SMTP      Response: 250 Ok

⊞ Frame 25 (60 bytes on wire, 60 bytes captured)
⊞ Ethernet II, Src: 00:0a:f4:32:b7:ff, Dst: 00:60:08:82:d2:19
⊟ Internet Protocol, Src Addr: 202.108.44.206 (202.108.44.206), Dst Addr: 10.214.6.15 (10.214.6.15)
    Version: 4
    Header length: 20 bytes
  ⊟ Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
      0000 00.. = Differentiated Services Codepoint: Default (0x00)
      .... ..0. = ECN-Capable Transport (ECT): 0
      .... ...0 = ECN-CE: 0
    Total Length: 40
    Identification: 0x3569 (13673)
  ⊟ Flags: 0x04 (Don't Fragment)
      0... = Reserved bit: Not set
      .1.. = Don't fragment: Set
      ..0. = More fragments: Not set
    Fragment offset: 0
    Time to live: 51
    Protocol: TCP (0x06)
    Header checksum: 0x0a48 (correct)
    Source: 202.108.44.206 (202.108.44.206)
    Destination: 10.214.6.15 (10.214.6.15)
⊟ Transmission Control Protocol, Src Port: smtp (25), Dst Port: 1686 (1686), Seq: 227, Ack: 91, Len: 0
    Source port: smtp (25)
    Destination port: 1686 (1686)
    Sequence number: 227    (relative sequence number)
    Acknowledgement number: 91    (relative ack number)
    Header length: 20 bytes
  ⊟ Flags: 0x0010 (ACK)
      0... .... = Congestion Window Reduced (CWR): Not set
      .0.. .... = ECN-Echo: Not set
      ..0. .... = Urgent: Not set
      ...1 .... = Acknowledgment: Set
      .... 0... = Push: Not set
      .... .0.. = Reset: Not set
      .... ..0. = Syn: Not set
      .... ...0 = Fin: Not set
    Window size: 5840
    Checksum: 0x8146 (correct)
  ⊟ [SEQ/ACK analysis]
      [This is an ACK to the segment in frame: 24]
      [The RTT to ACK the segment was: 0.050692000 seconds]

0000  00 60 08 82 d2 19 00 0a  f4 32 b7 ff 08 00 45 00   .`...... .2....E.
0010  00 28 35 69 40 00 33 06  0a 48 ca 6c 2c ce 0a d6   .(5i@.3. .H.l,...
0020  06 0f 00 19 06 96 20 09  7f 7e 99 a6 cf c1 50 10   ...... . .~....P.
0030  16 d0 81 46 00 00 00 00  00 00 00 00               ...F.... ....

File: mail smtp 5682 bytes 00:00:20                P: 49 D: 49 M: 0
```

图　9-40

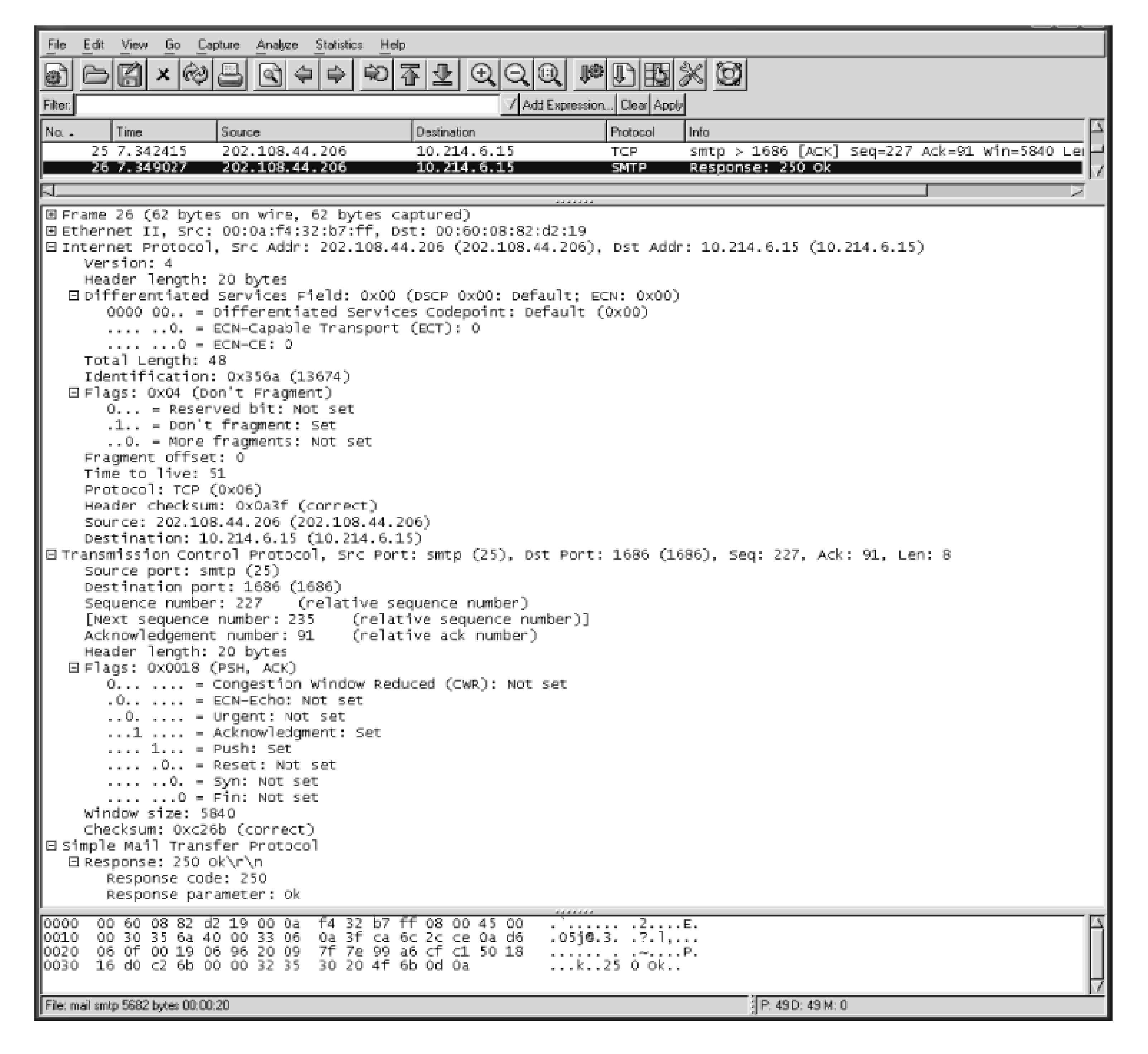

File Edit View Go Capture Analyze Statistics Help
Filter: Add Expression... Clear Apply
No. Time Source Destination Protocol Info
25 7.342415 202.108.44.206 10.214.6.15 TCP smtp > 1686 [ACK] Seq=227 Ack=91 Win=5840 Len
26 7.349027 202.108.44.206 10.214.6.15 SMTP Response: 250 Ok
Frame 26 (62 bytes on wire, 62 bytes captured)
Ethernet II, Src: 00:0a:f4:32:b7:ff, Dst: 00:60:08:82:d2:19
Internet Protocol, Src Addr: 202.108.44.206 (202.108.44.206), Dst Addr: 10.214.6.15 (10.214.6.15)
Version: 4
Header length: 20 bytes
Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
0000 00.. = Differentiated Services Codepoint: Default (0x00)
.... ..0. = ECN-Capable Transport (ECT): 0
.... ...0 = ECN-CE: 0
Total Length: 48
Identification: 0x356a (13674)
Flags: 0x04 (Don't Fragment)
0... = Reserved bit: Not set
.1.. = Don't fragment: Set
..0. = More fragments: Not set
Fragment offset: 0
Time to live: 51
Protocol: TCP (0x06)
Header checksum: 0x0a3f (correct)
Source: 202.108.44.206 (202.108.44.206)
Destination: 10.214.6.15 (10.214.6.15)
Transmission Control Protocol, Src Port: smtp (25), Dst Port: 1686 (1686), Seq: 227, Ack: 91, Len: 8
Source port: smtp (25)
Destination port: 1686 (1686)
Sequence number: 227 (relative sequence number)
[Next sequence number: 235 (relative sequence number)]
Acknowledgement number: 91 (relative ack number)
Header length: 20 bytes
Flags: 0x0018 (PSH, ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 1... = Push: Set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 5840
Checksum: 0xc26b (correct)
Simple Mail Transfer Protocol
Response: 250 Ok\r\n
Response code: 250
Response parameter: Ok
0000 00 60 08 82 d2 19 00 0a f4 32 b7 ff 08 00 45 00
0010 00 30 35 6a 40 00 33 06 0a 3f ca 6c 2c ce 0a d6
0020 06 0f 00 19 06 96 20 09 7f 7e 99 a6 cf c1 50 18
0030 16 d0 c2 6b 00 00 32 35 30 20 4f 6b 0d 0a
File: mail smtp 5682 bytes 00:00:20
P: 49 D: 49 M: 0

图 9-41

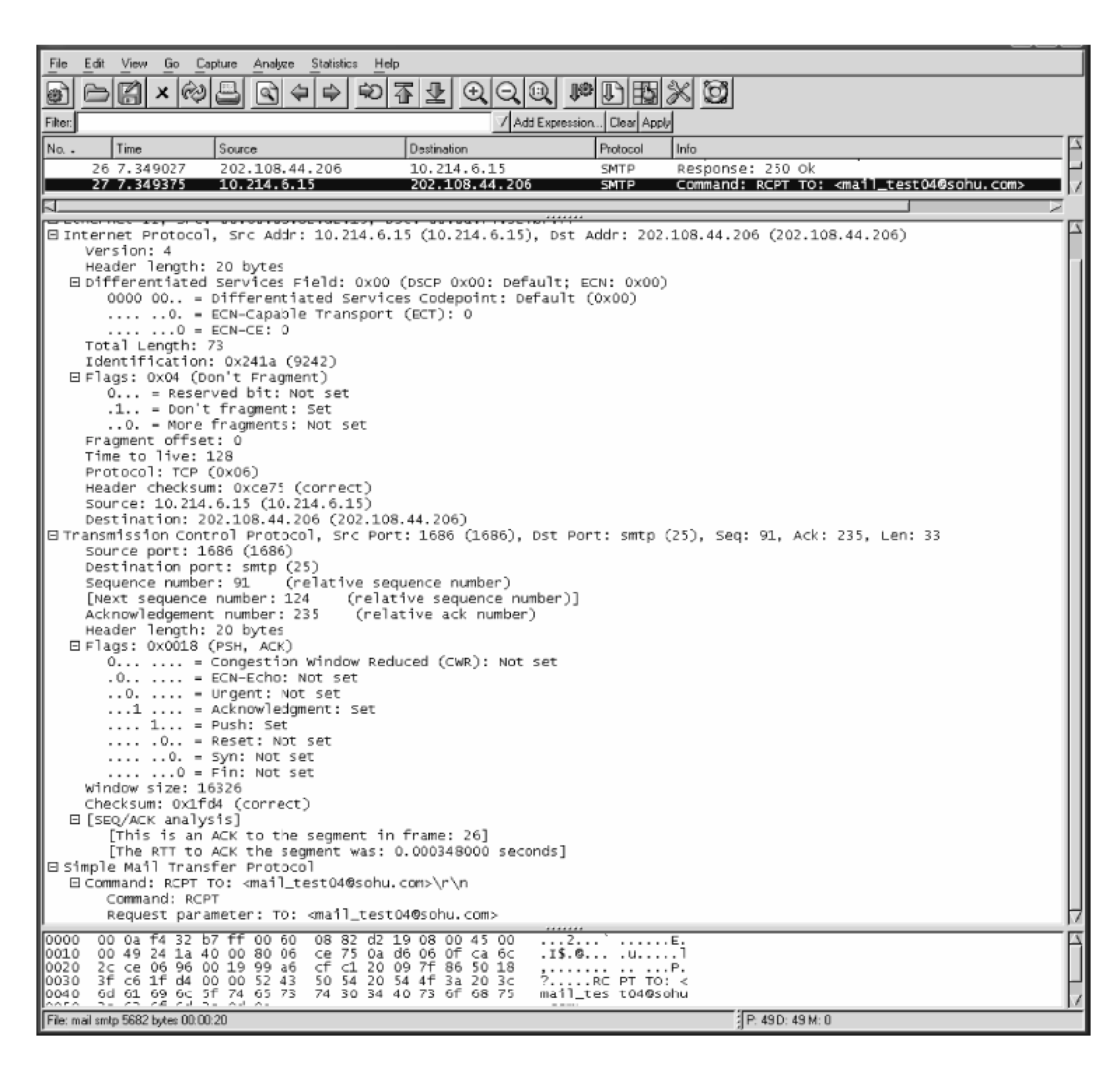

File Edit View Go Capture Analyze Statistics Help
Filter: Add Expression... Clear Apply
No. Time Source Destination Protocol Info
26 7.349027 202.108.44.206 10.214.6.15 SMTP Response: 250 Ok
27 7.349375 10.214.6.15 202.108.44.206 SMTP Command: RCPT TO: <mail_test04@sohu.com>
Internet Protocol, Src Addr: 10.214.6.15 (10.214.6.15), Dst Addr: 202.108.44.206 (202.108.44.206)
Version: 4
Header length: 20 bytes
Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
0000 00.. = Differentiated Services Codepoint: Default (0x00)
.... ..0. = ECN-Capable Transport (ECT): 0
.... ...0 = ECN-CE: 0
Total Length: 73
Identification: 0x241a (9242)
Flags: 0x04 (Don't Fragment)
0... = Reserved bit: Not set
.1.. = Don't fragment: Set
..0. = More fragments: Not set
Fragment offset: 0
Time to live: 128
Protocol: TCP (0x06)
Header checksum: 0xce75 (correct)
Source: 10.214.6.15 (10.214.6.15)
Destination: 202.108.44.206 (202.108.44.206)
Transmission Control Protocol, Src Port: 1686 (1686), Dst Port: smtp (25), Seq: 91, Ack: 235, Len: 33
Source port: 1686 (1686)
Destination port: smtp (25)
Sequence number: 91 (relative sequence number)
[Next sequence number: 124 (relative sequence number)]
Acknowledgement number: 235 (relative ack number)
Header length: 20 bytes
Flags: 0x0018 (PSH, ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 1... = Push: Set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 16326
Checksum: 0x1fd4 (correct)
[SEQ/ACK analysis]
[This is an ACK to the segment in frame: 26]
[The RTT to ACK the segment was: 0.000348000 seconds]
Simple Mail Transfer Protocol
Command: RCPT TO: <mail_test04@sohu.com>\r\n
Command: RCPT
Request parameter: TO: <mail_test04@sohu.com>
0000 00 0a f4 32 b7 ff 00 60 08 82 d2 19 08 00 45 00
0010 00 49 24 1a 40 00 80 06 ce 75 0a d6 06 0f ca 6c
0020 2c ce 06 96 00 19 99 a6 cf c1 20 09 7f 86 50 18
0030 3f c6 1f d4 00 00 52 43 50 54 20 54 4f 3a 20 3c
File: mail smtp 5682 bytes 00:00:20
P: 49 D: 49 M: 0

图 9-42

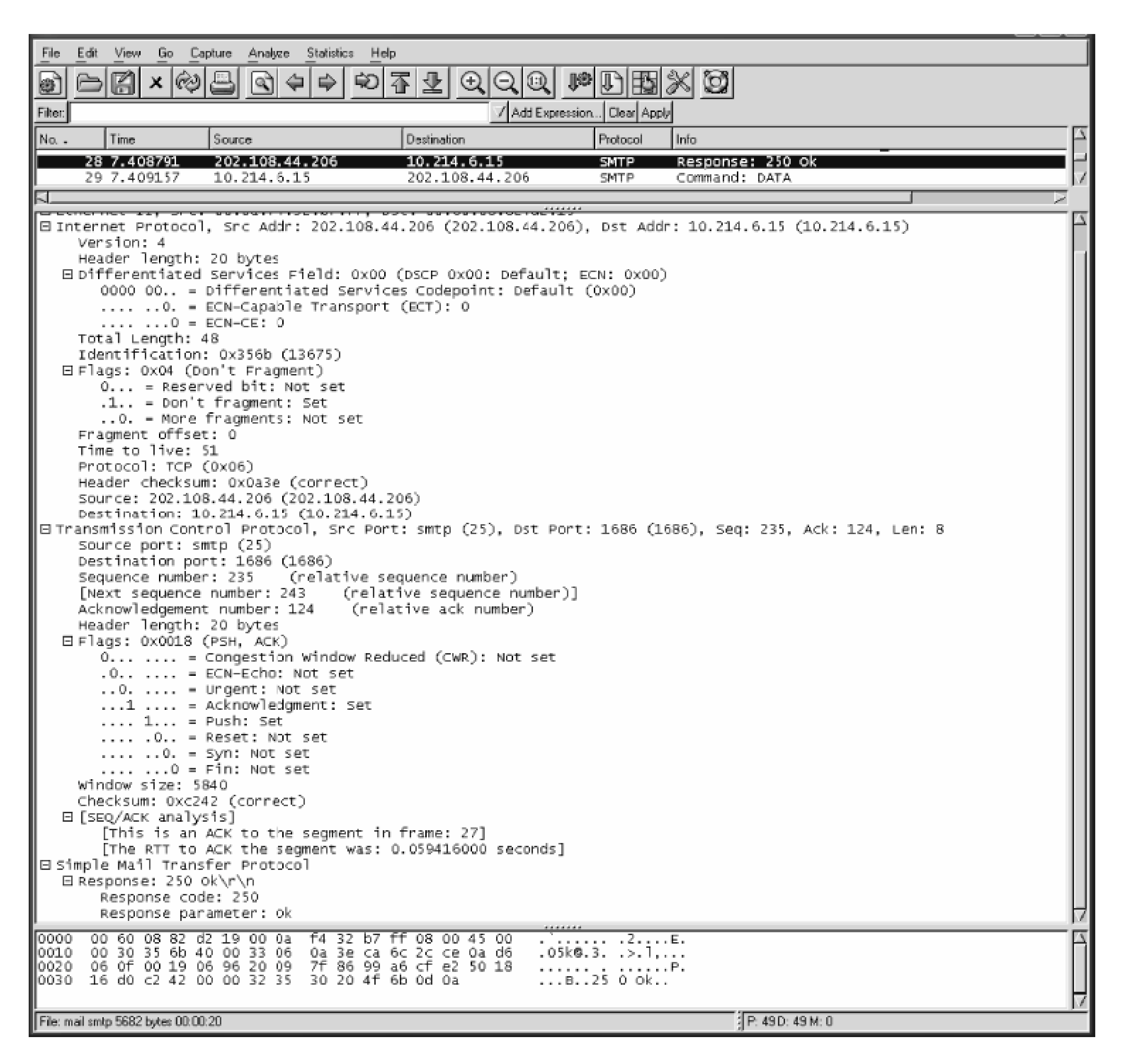



图　9-43

客户端发送 DATA 命令，表示将要发送邮件正文了，如图 9-44 所示。

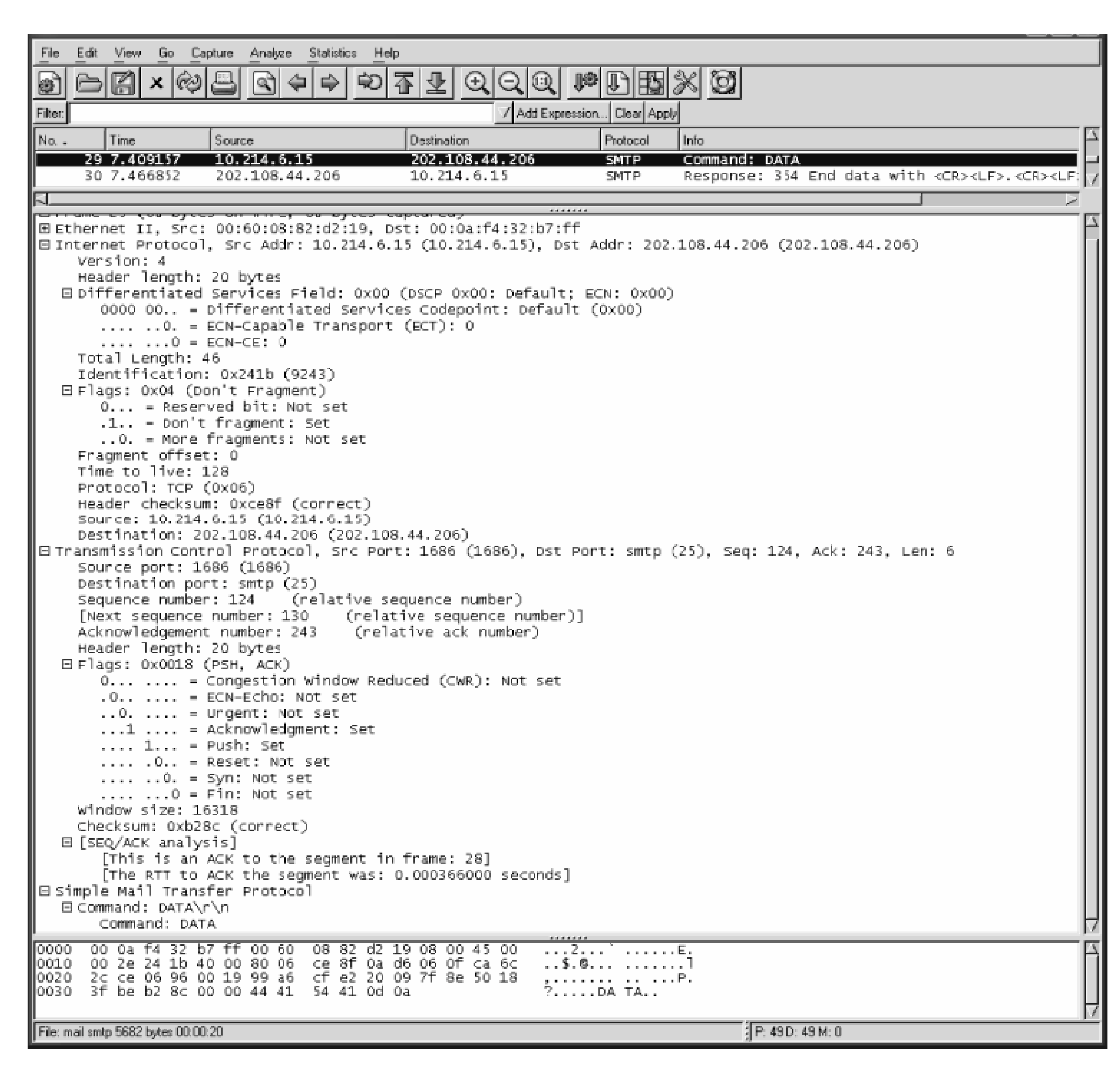



图　9-44

服务器对 DATA 命令返回一个应答，应答码为 354，表示服务器准备接受数据，数据包文以＜CR＞＜LF＞．＜CR＞＜LF＞的行结束，如图 9-45 所示。

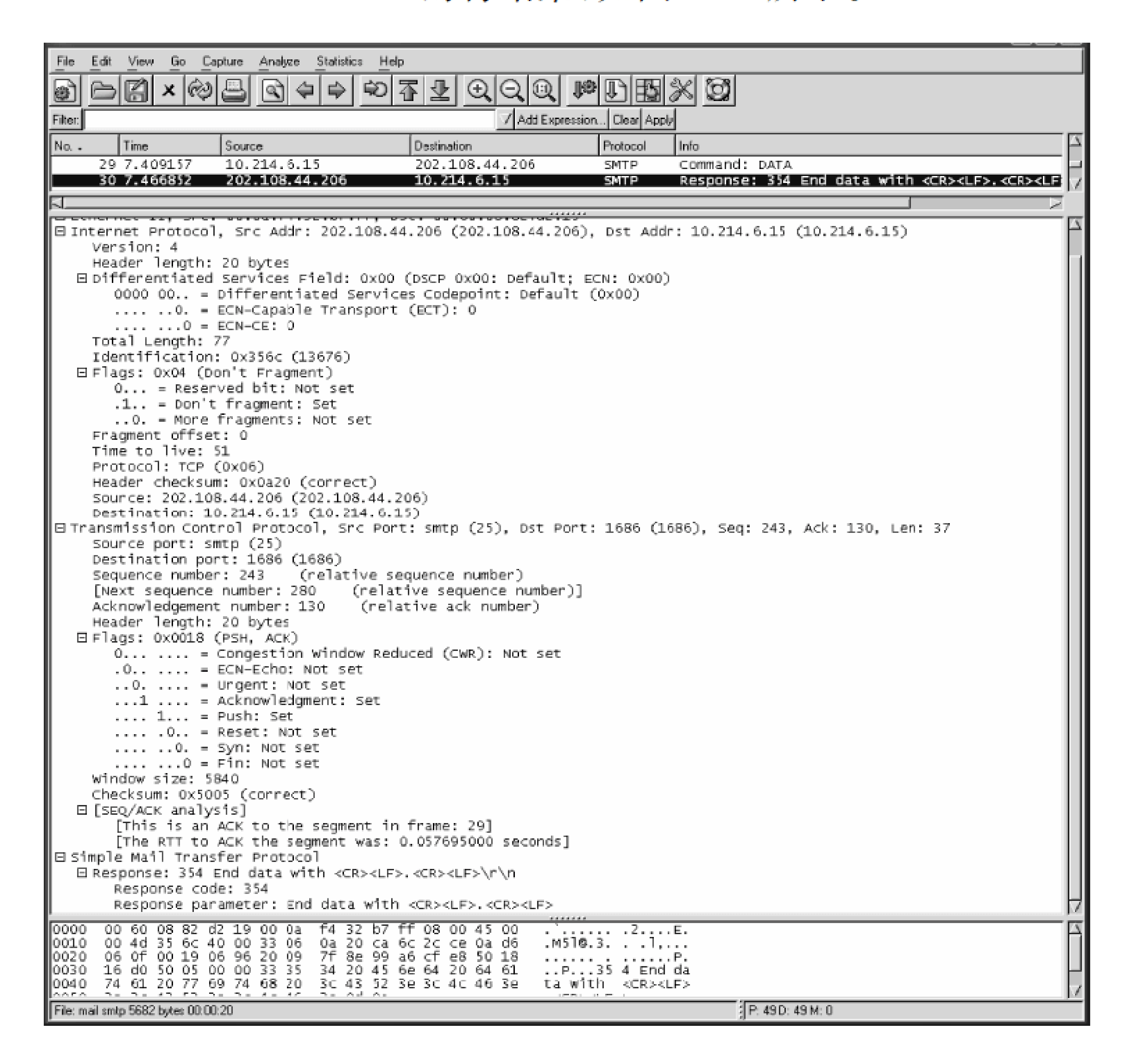



图 9-45

客户端传送邮件正文，如图 9-46 所示，客户端用连续的行发送报文内容，每一行的行结束用＜CRLF＞终止。

如图 9-46 所示，传送的报文中，MIME-Version 以前的部分为电子邮件首部，MIME-Version 开始为 MIME 首部。MIME(Multipurpose Internet Email Extension，多用途因特网邮件扩充协议)允许非 ASCII 数据能够通过电子邮件传送。由于发送电子邮件只能使用 NVT ASCII 格式，那么邮件中的其他格式如图像、mp3、中文等都无法发送。因此，在 1992 年的时候人们利用 MIME，在发送方把非 ACSII 数据转换为 NVT ACSII 数据，接收方再把收到的数据还原。MIME 定义了 5 种首部，用来加在原始的邮件首部来定义参数的转换：MIME-version、Content-Type 、Content-Transfer-Encoding、Content-Id 和 Content-Description。MIME 后来也应用了浏览器，为了让浏览器知道接收到的信息哪些是 MP3 文件，哪些是 Shockwave 文件，服务器需要将发送的多媒体数据的类型告诉浏览器，也就是说明该多媒体数据的 MIME 类型，服务器将 MIME 标志符放入传送的数据中来告诉浏览器使用哪种插件读取相关文件。

如图 9-47 所示，服务器发送 TCP 确认，对收到的数据进行确认。

客户端发送文件结束信息 EOM(End Of Message)，见图 9-48。0d 0a 2e 0d 0a，2e 对应的 ASCII 字符是"．"，十六进制数 0d 对应的 ASCII 字符是"CR"，意思是回车，十六机制数 0a 对应的 ASCII 字符"LF"意思是"换行"。

服务器端发送 TCP 确认，如图 9-49 所示。

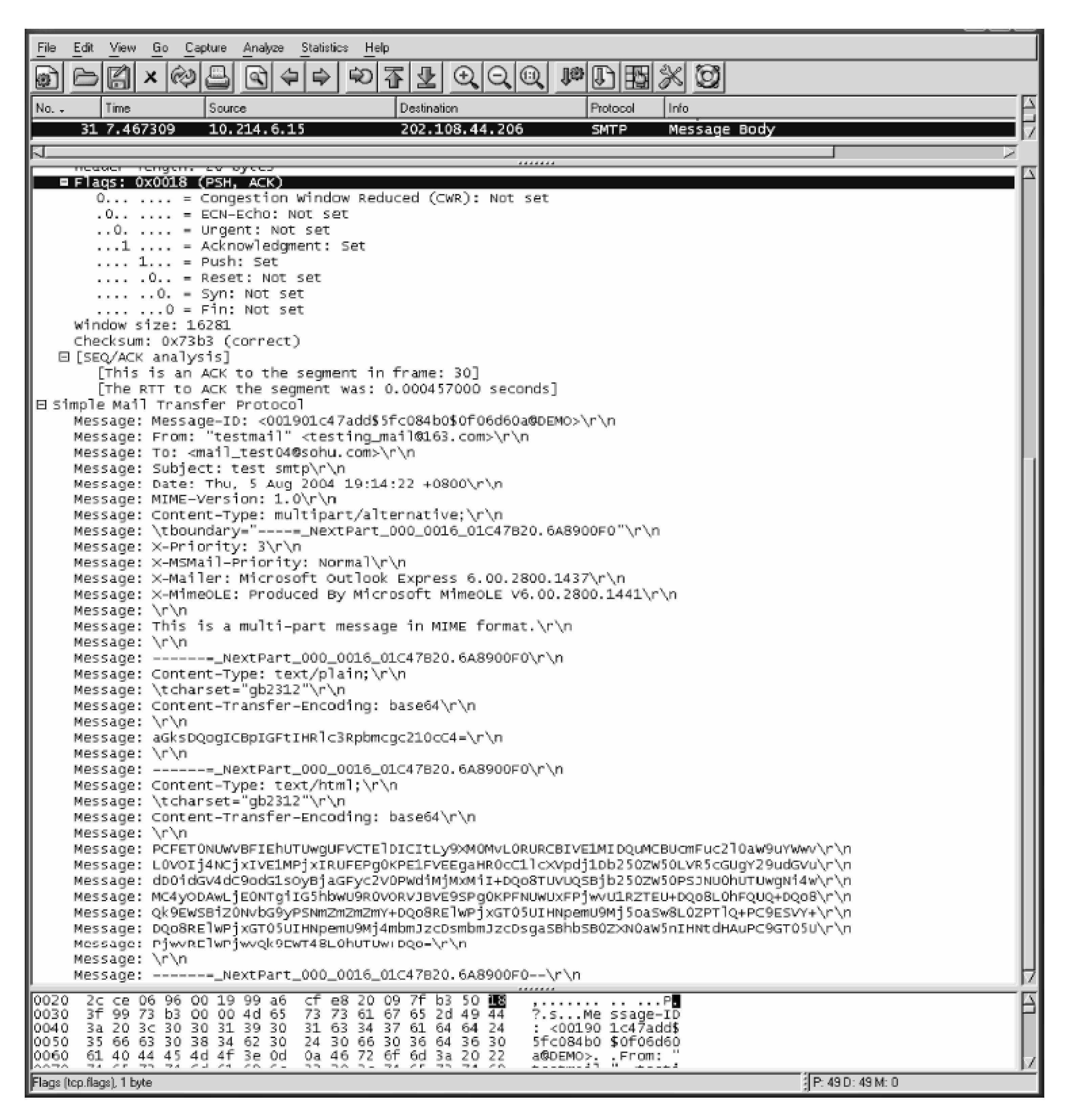

File Edit View Go Capture Analyze Statistics Help
No. Time Source Destination Protocol Info
31 7.467309 10.214.6.15 202.108.44.206 SMTP Message Body
Flags: 0x0018 (PSH, ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 1... = Push: Set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 16281
Checksum: 0x73b3 (correct)
[SEQ/ACK analysis]
[This is an ACK to the segment in frame: 30]
[The RTT to ACK the segment was: 0.000457000 seconds]
Simple Mail Transfer Protocol
Message: Message-ID: <001901c47add$5fc084b0$0f06d60a@DEMO>\r\n
Message: From: "testmail" <testing_mail@163.com>\r\n
Message: To: <mail_test04@sohu.com>\r\n
Message: Subject: test smtp\r\n
Message: Date: Thu, 5 Aug 2004 19:14:22 +0800\r\n
Message: MIME-Version: 1.0\r\n
Message: Content-Type: multipart/alternative;\r\n
Message: \tboundary="----=_NextPart_000_0016_01C47B20.6A8900F0"\r\n
Message: X-Priority: 3\r\n
Message: X-MSMail-Priority: Normal\r\n
Message: X-Mailer: Microsoft Outlook Express 6.00.2800.1437\r\n
Message: X-MimeOLE: Produced By Microsoft MimeOLE V6.00.2800.1441\r\n
Message: \r\n
Message: This is a multi-part message in MIME format.\r\n
Message: \r\n
Message: ------=_NextPart_000_0016_01C47B20.6A8900F0\r\n
Message: Content-Type: text/plain;\r\n
Message: \tcharset="gb2312"\r\n
Message: Content-Transfer-Encoding: base64\r\n
Message: \r\n
Message: aGksDQogICBpIGFtIHRlc3Rpbmcgc210cC4=\r\n
Message: \r\n
Message: ------=_NextPart_000_0016_01C47B20.6A8900F0\r\n
Message: Content-Type: text/html;\r\n
Message: \tcharset="gb2312"\r\n
Message: Content-Transfer-Encoding: base64\r\n
Message: \r\n
Message: PCFET0NUWVBFIEhUTUwgUFVCTElDICItLy9XM0MvL0RURCBIVE1MIDQuMCBUcmFuc2l0aW9uYWwv\r\n
Message: L0VOIj4NCjxIVE1MPjxIRUFEPg0KPE1FVEEgaHR0cC1lcXVpdj1Db250ZW50LVR5cGUgY29udGVu\r\n
Message: dD0idGV4dC9odG1sOyBjaGFyc2V0PWdiMjMxMiI+DQo8TUVUQSBjb250ZW50PSJNU0hUTUwgNi4w\r\n
Message: MC4yODAwLjE0NTgiIG5hbWU9R0VORVJBVE9SPg0KPFNUWUxFPjwvU1RZTEU+DQo8L0hFQUQ+DQo8\r\n
Message: Qk9EWSBiZ0NvbG9yPSNmZmZmZmY+DQo8RElWPjxGT05UIHNpemU9Mj5oaSw8L0ZPTlQ+PC9ESVY+\r\n
Message: DQo8RElWPjxGT05UIHNpemU9Mj4mbmJzcDsmbmJzcDsgaSBhbSB0ZXN0aW5nIHNtdHAuPC9GT05U\r\n
Message: PjwvRElWPjwvQk9EWT48L0hUTUw+DQo=\r\n
Message: \r\n
Message: ------=_NextPart_000_0016_01C47B20.6A8900F0--\r\n
0020 2c ce 06 96 00 19 99 a6 cf e8 20 09 7f b3 50 18 ,....... .. ...P.
0030 3f 99 73 b3 00 00 4d 65 73 73 61 67 65 2d 49 44 ?.s...Me ssage-ID
0040 3a 20 3c 30 30 31 39 30 31 63 34 37 61 64 64 24 : <00190 1c47add$
0050 35 66 63 30 38 34 62 30 24 30 66 30 36 64 36 30 5fc084b0 $0f06d60
0060 61 40 44 45 4d 4f 3e 0d 0a 46 72 6f 6d 3a 20 22 a@DEMO>. .From: "
Flags (tcp.flags), 1 byte
P: 49 D: 49 M: 0

图　9-46

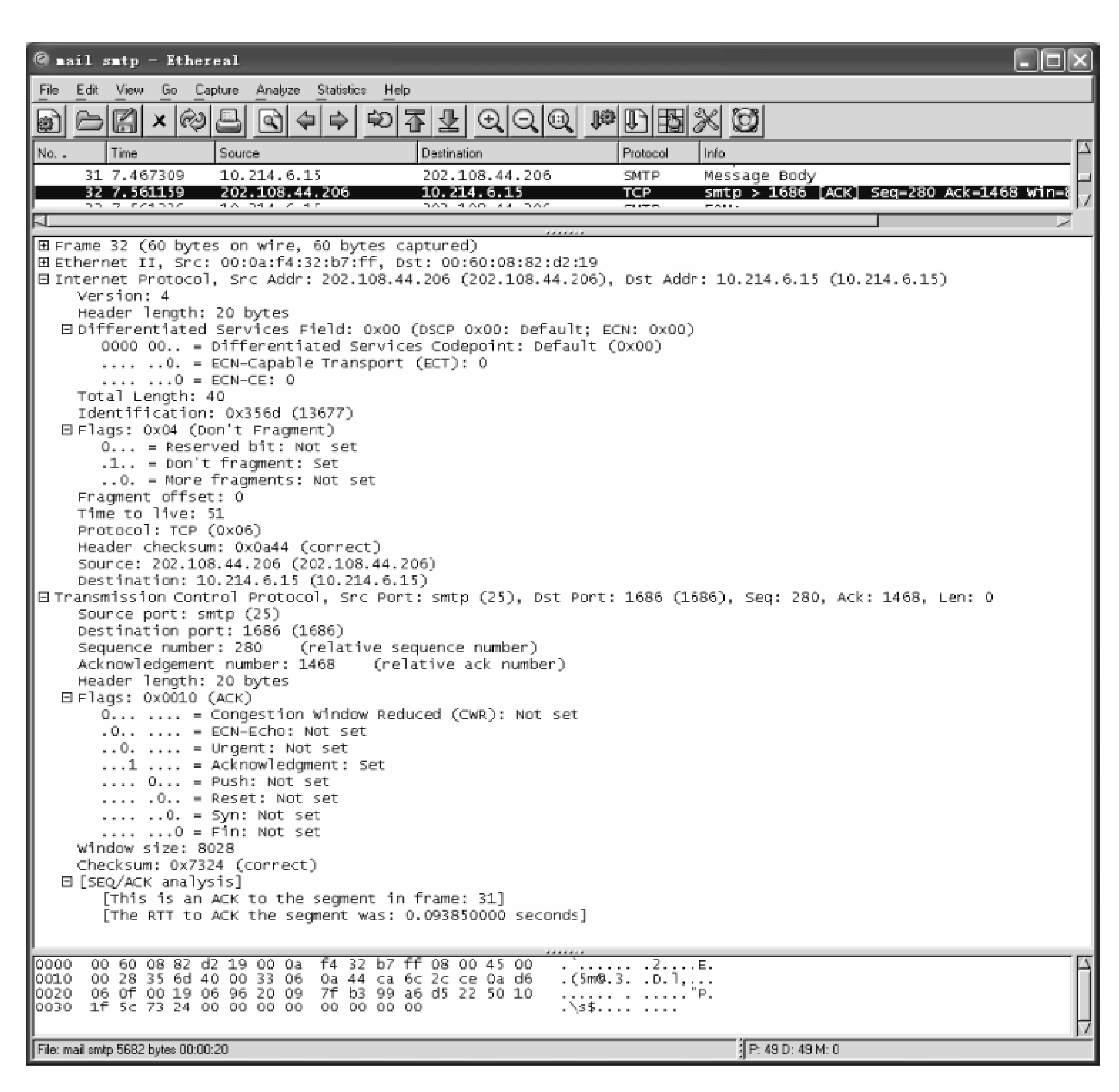

mail smtp - Ethereal
File Edit View Go Capture Analyze Statistics Help
No. Time Source Destination Protocol Info
31 7.467309 10.214.6.15 202.108.44.206 SMTP Message Body
32 7.561159 202.108.44.206 10.214.6.15 TCP smtp > 1686 [ACK] Seq=280 Ack=1468 Win=
Frame 32 (60 bytes on wire, 60 bytes captured)
Ethernet II, Src: 00:0a:f4:32:b7:ff, Dst: 00:60:08:82:d2:19
Internet Protocol, Src Addr: 202.108.44.206 (202.108.44.206), Dst Addr: 10.214.6.15 (10.214.6.15)
Version: 4
Header length: 20 bytes
Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
0000 00.. = Differentiated Services Codepoint: Default (0x00)
.... ..0. = ECN-Capable Transport (ECT): 0
.... ...0 = ECN-CE: 0
Total Length: 40
Identification: 0x356d (13677)
Flags: 0x04 (Don't Fragment)
0... = Reserved bit: Not set
.1.. = Don't fragment: Set
..0. = More fragments: Not set
Fragment offset: 0
Time to live: 51
Protocol: TCP (0x06)
Header checksum: 0x0a44 (correct)
Source: 202.108.44.206 (202.108.44.206)
Destination: 10.214.6.15 (10.214.6.15)
Transmission Control Protocol, Src Port: smtp (25), Dst Port: 1686 (1686), Seq: 280, Ack: 1468, Len: 0
Source port: smtp (25)
Destination port: 1686 (1686)
Sequence number: 280 (relative sequence number)
Acknowledgement number: 1468 (relative ack number)
Header length: 20 bytes
Flags: 0x0010 (ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 0... = Push: Not set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 8028
Checksum: 0x7324 (correct)
[SEQ/ACK analysis]
[This is an ACK to the segment in frame: 31]
[The RTT to ACK the segment was: 0.093850000 seconds]
0000 00 60 08 82 d2 19 00 0a f4 32 b7 ff 08 00 45 00 .`...... .2....E.
0010 00 28 35 6d 40 00 33 06 0a 44 ca 6c 2c ce 0a d6 .(5m@.3. .D.l,...
0020 06 0f 00 19 06 96 20 09 7f b3 99 a6 d5 22 50 10 ...... . ....."P.
0030 1f 5c 73 24 00 00 00 00 00 00 00 00 .\s$.... ....
File: mail smtp 5682 bytes 00:00:20
P: 49 D: 49 M: 0

图　9-47

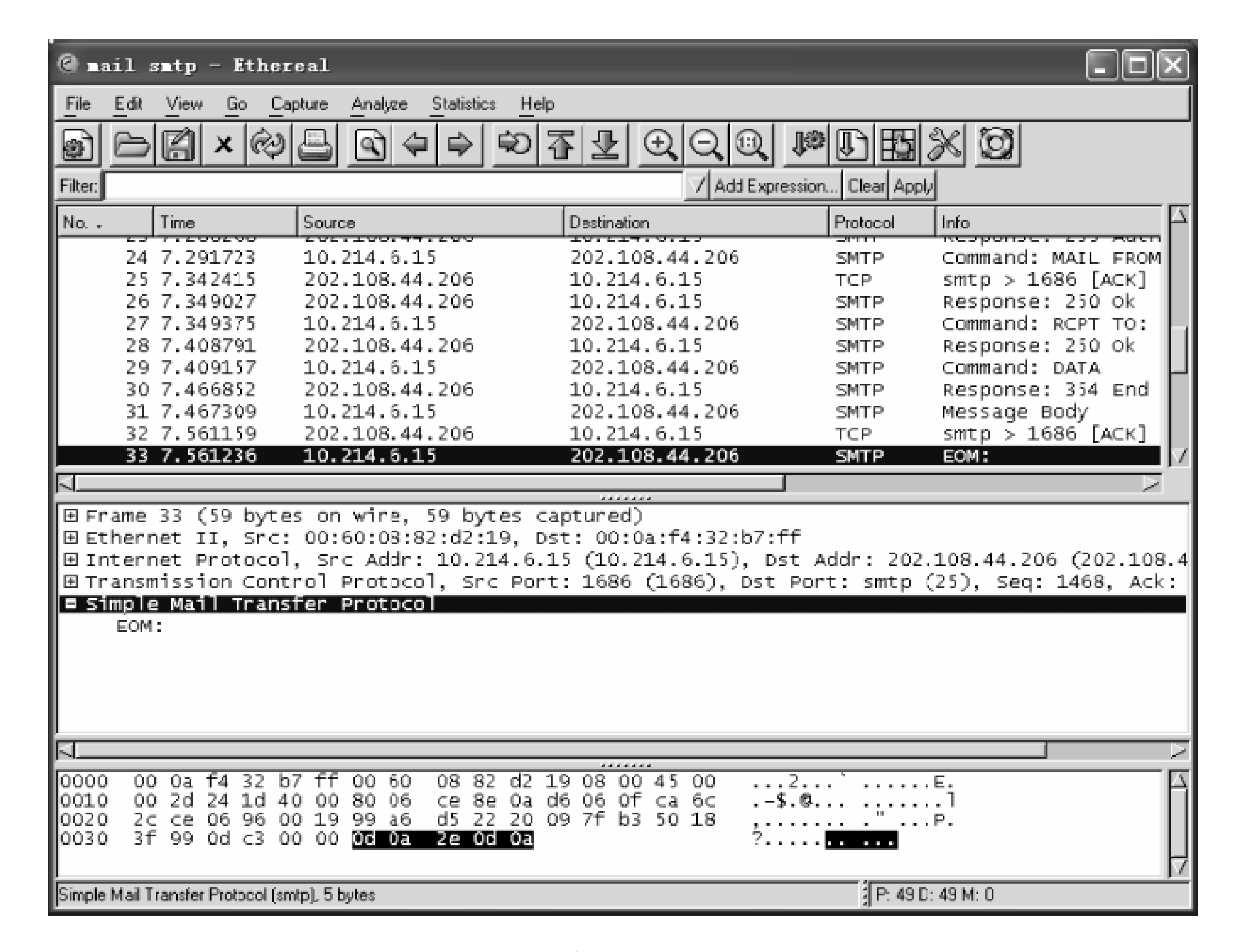



图 9-48

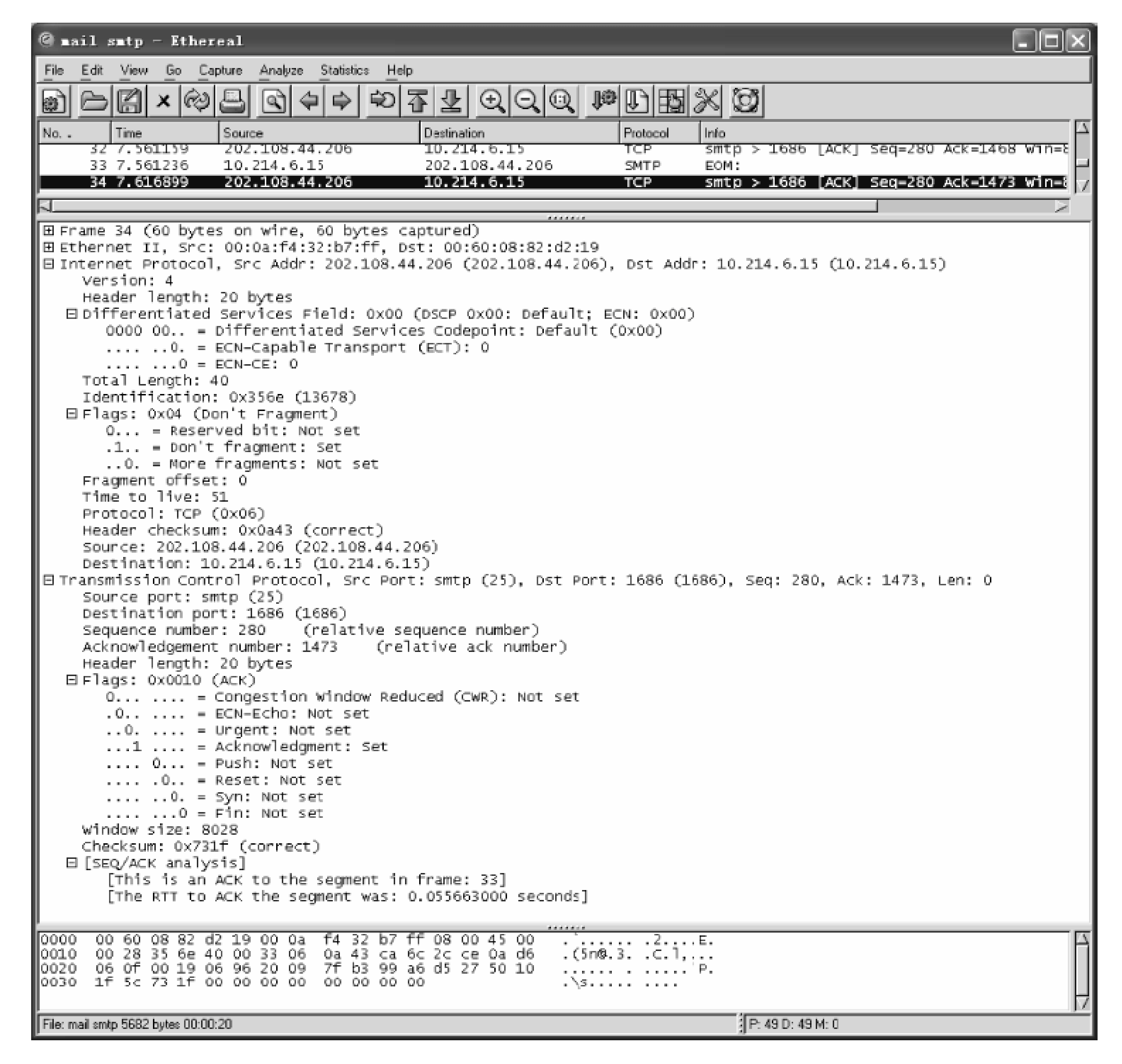



图 9-49

服务器返回 SMTP 响应，应答码为 250，表示操作成功，如图 9-50 所示。到此一封邮件已经发送成功，你可以发送下一封邮件。

客户端用 QUIT 命令来结束连接，如图 9-51 所示。

服务器返回 TCP 确认，如图 9-52 所示。

服务器端返回 SMTP 响应，应答码为 221，表示服务关闭，如图 9-53 所示。

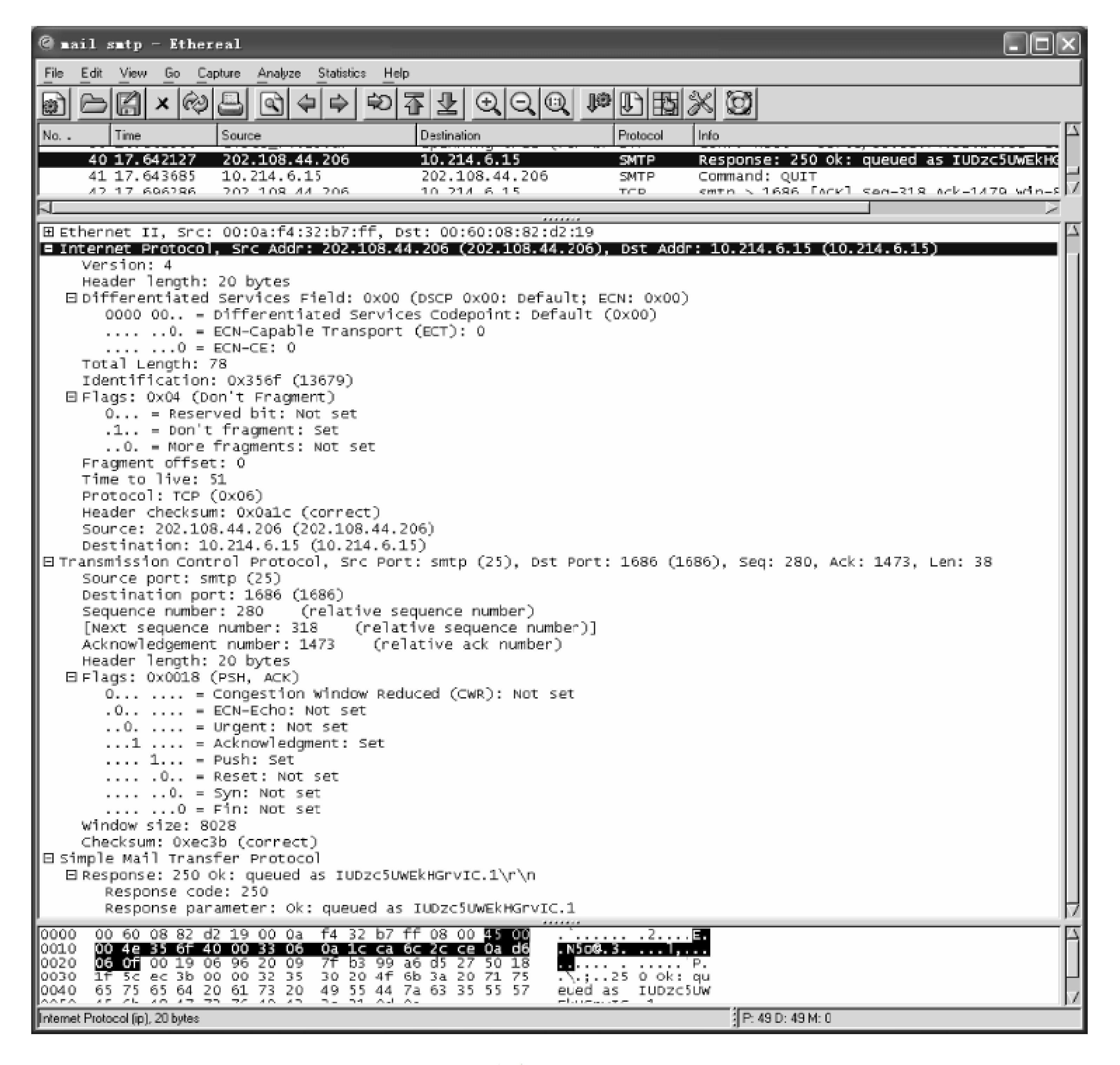

mail smtp - Ethereal
File Edit View Go Capture Analyze Statistics Help
No. . Time Source Destination Protocol Info
40 17.642127 202.108.44.206 10.214.6.15 SMTP Response: 250 Ok: queued as IUDzc5UWEkHG
41 17.643685 10.214.6.15 202.108.44.206 SMTP Command: QUIT
Ethernet II, Src: 00:0a:f4:32:b7:ff, Dst: 00:60:08:82:d2:19
Internet Protocol, Src Addr: 202.108.44.206 (202.108.44.206), Dst Addr: 10.214.6.15 (10.214.6.15)
Version: 4
Header length: 20 bytes
Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
0000 00.. = Differentiated Services Codepoint: Default (0x00)
.... ..0. = ECN-Capable Transport (ECT): 0
.... ...0 = ECN-CE: 0
Total Length: 78
Identification: 0x356f (13679)
Flags: 0x04 (Don't Fragment)
0... = Reserved bit: Not set
.1.. = Don't fragment: Set
..0. = More fragments: Not set
Fragment offset: 0
Time to live: 51
Protocol: TCP (0x06)
Header checksum: 0x0a1c (correct)
Source: 202.108.44.206 (202.108.44.206)
Destination: 10.214.6.15 (10.214.6.15)
Transmission Control Protocol, Src Port: smtp (25), Dst Port: 1686 (1686), Seq: 280, Ack: 1473, Len: 38
Source port: smtp (25)
Destination port: 1686 (1686)
Sequence number: 280 (relative sequence number)
[Next sequence number: 318 (relative sequence number)]
Acknowledgement number: 1473 (relative ack number)
Header length: 20 bytes
Flags: 0x0018 (PSH, ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 1... = Push: Set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 8028
Checksum: 0xec3b (correct)
Simple Mail Transfer Protocol
Response: 250 Ok: queued as IUDzc5UWEkHGrvIC.1\r\n
Response code: 250
Response parameter: Ok: queued as IUDzc5UWEkHGrvIC.1
Internet Protocol (ip), 20 bytes
P: 49 D: 49 M: 0

图　9-50

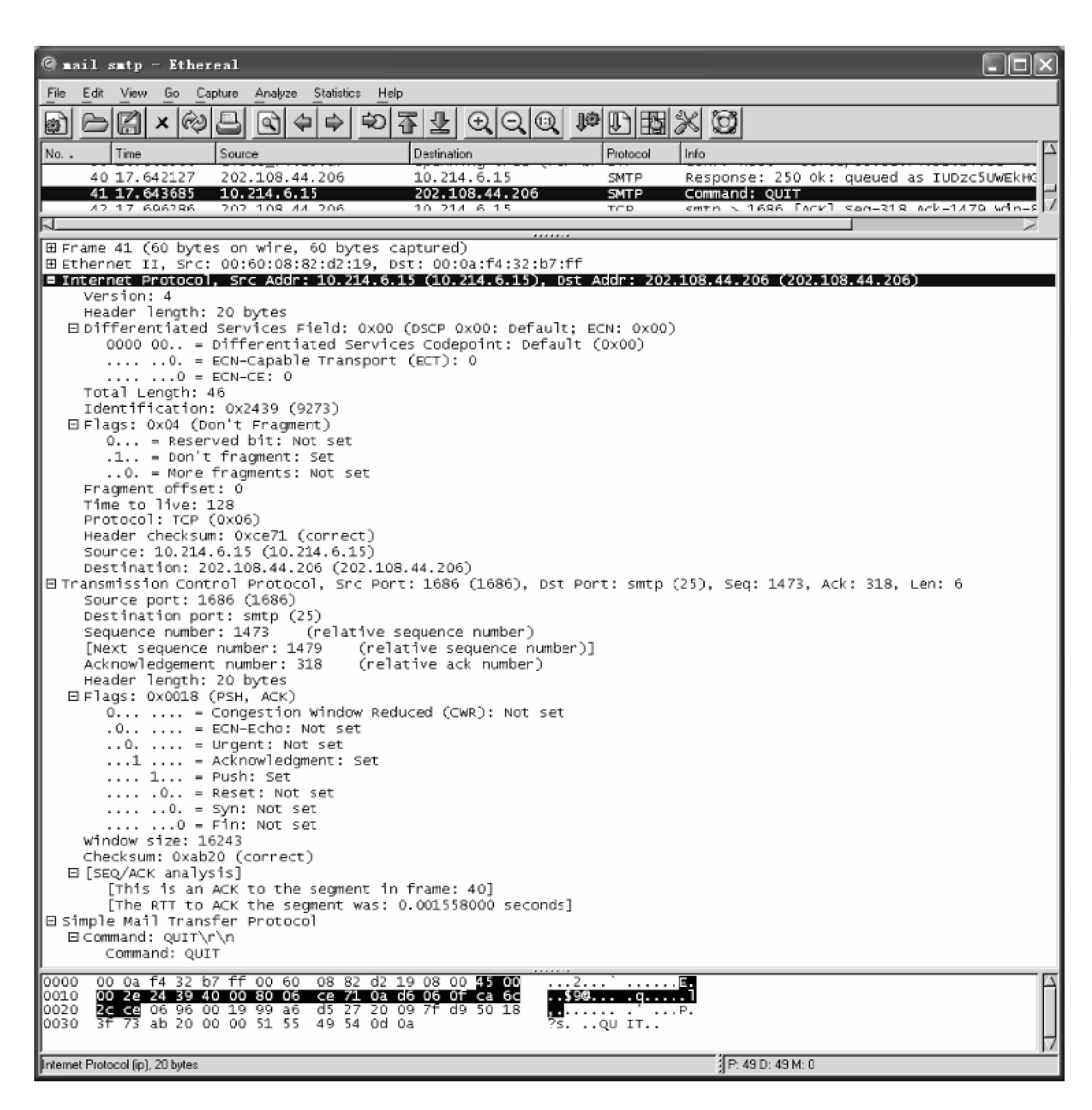

mail smtp - Ethereal
File Edit View Go Capture Analyze Statistics Help
No. . Time Source Destination Protocol Info
40 17.642127 202.108.44.206 10.214.6.15 SMTP Response: 250 Ok: queued as IUDzc5UWEkHG
41 17.643685 10.214.6.15 202.108.44.206 SMTP Command: QUIT
Frame 41 (60 bytes on wire, 60 bytes captured)
Ethernet II, Src: 00:60:08:82:d2:19, Dst: 00:0a:f4:32:b7:ff
Internet Protocol, Src Addr: 10.214.6.15 (10.214.6.15), Dst Addr: 202.108.44.206 (202.108.44.206)
Version: 4
Header length: 20 bytes
Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
0000 00.. = Differentiated Services Codepoint: Default (0x00)
.... ..0. = ECN-Capable Transport (ECT): 0
.... ...0 = ECN-CE: 0
Total Length: 46
Identification: 0x2439 (9273)
Flags: 0x04 (Don't Fragment)
0... = Reserved bit: Not set
.1.. = Don't fragment: Set
..0. = More fragments: Not set
Fragment offset: 0
Time to live: 128
Protocol: TCP (0x06)
Header checksum: 0xce71 (correct)
Source: 10.214.6.15 (10.214.6.15)
Destination: 202.108.44.206 (202.108.44.206)
Transmission Control Protocol, Src Port: 1686 (1686), Dst Port: smtp (25), Seq: 1473, Ack: 318, Len: 6
Source port: 1686 (1686)
Destination port: smtp (25)
Sequence number: 1473 (relative sequence number)
[Next sequence number: 1479 (relative sequence number)]
Acknowledgement number: 318 (relative ack number)
Header length: 20 bytes
Flags: 0x0018 (PSH, ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 1... = Push: Set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 16243
Checksum: 0xab20 (correct)
[SEQ/ACK analysis]
[This is an ACK to the segment in frame: 40]
[The RTT to ACK the segment was: 0.001558000 seconds]
Simple Mail Transfer Protocol
Command: QUIT\r\n
Command: QUIT
Internet Protocol (ip), 20 bytes
P: 49 D: 49 M: 0

图　9-51

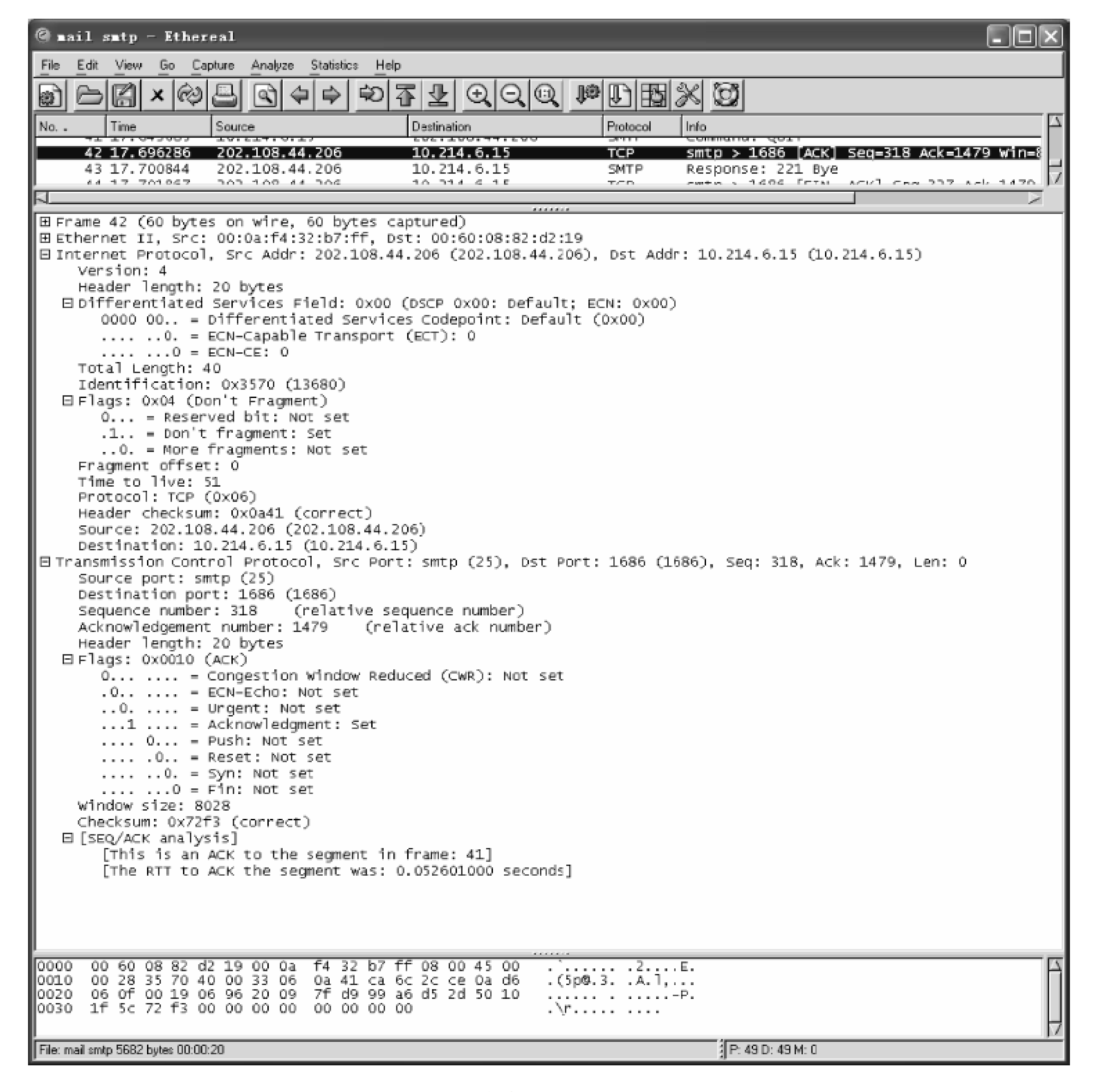

mail smtp - Ethereal
File Edit View Go Capture Analyze Statistics Help
No. . Time Source Destination Protocol Info
42 17.696286 202.108.44.206 10.214.6.15 TCP smtp > 1686 [ACK] Seq=318 Ack=1479 Win=8
43 17.700844 202.108.44.206 10.214.6.15 SMTP Response: 221 Bye
Frame 42 (60 bytes on wire, 60 bytes captured)
Ethernet II, Src: 00:0a:f4:32:b7:ff, Dst: 00:60:08:82:d2:19
Internet Protocol, Src Addr: 202.108.44.206 (202.108.44.206), Dst Addr: 10.214.6.15 (10.214.6.15)
Version: 4
Header length: 20 bytes
Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
0000 00.. = Differentiated Services Codepoint: Default (0x00)
.... ..0. = ECN-Capable Transport (ECT): 0
.... ...0 = ECN-CE: 0
Total Length: 40
Identification: 0x3570 (13680)
Flags: 0x04 (Don't Fragment)
0... = Reserved bit: Not set
.1.. = Don't fragment: Set
..0. = More fragments: Not set
Fragment offset: 0
Time to live: 51
Protocol: TCP (0x06)
Header checksum: 0x0a41 (correct)
Source: 202.108.44.206 (202.108.44.206)
Destination: 10.214.6.15 (10.214.6.15)
Transmission Control Protocol, Src Port: smtp (25), Dst Port: 1686 (1686), Seq: 318, Ack: 1479, Len: 0
Source port: smtp (25)
Destination port: 1686 (1686)
Sequence number: 318 (relative sequence number)
Acknowledgement number: 1479 (relative ack number)
Header length: 20 bytes
Flags: 0x0010 (ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 0... = Push: Not set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 8028
Checksum: 0x72f3 (correct)
[SEQ/ACK analysis]
[This is an ACK to the segment in frame: 41]
[The RTT to ACK the segment was: 0.052601000 seconds]
0000 00 60 08 82 d2 19 00 0a f4 32 b7 ff 08 00 45 00
0010 00 28 35 70 40 00 33 06 0a 41 ca 6c 2c ce 0a d6
0020 06 0f 00 19 06 96 20 09 7f d9 99 a6 d5 2d 50 10
0030 1f 5c 72 f3 00 00 00 00 00 00 00 00
File: mail smtp 5682 bytes 00:00:20
P: 49 D: 49 M: 0

图 9-52

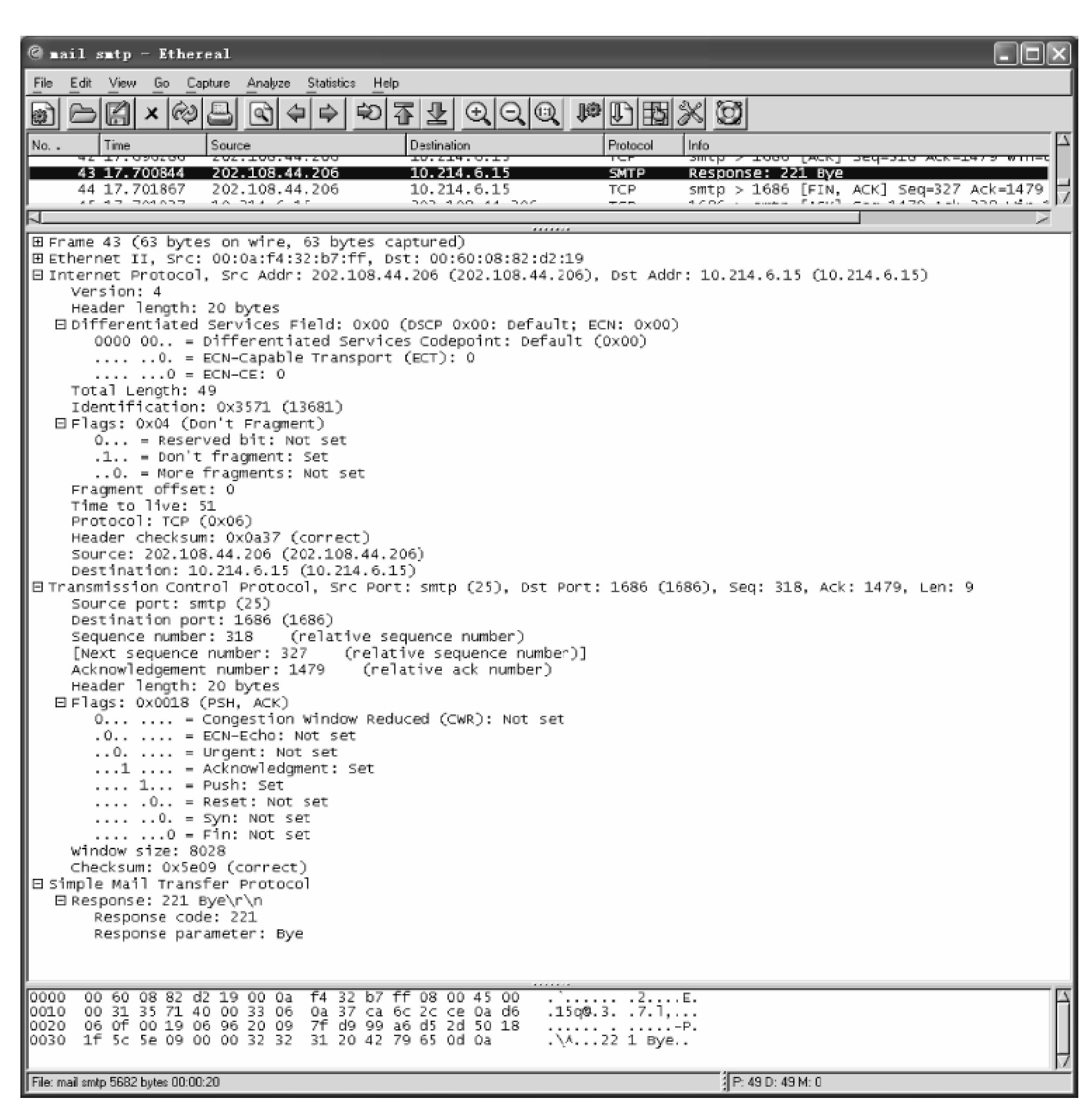

mail smtp - Ethereal
File Edit View Go Capture Analyze Statistics Help
No. . Time Source Destination Protocol Info
43 17.700844 202.108.44.206 10.214.6.15 SMTP Response: 221 Bye
44 17.701867 202.108.44.206 10.214.6.15 TCP smtp > 1686 [FIN, ACK] Seq=327 Ack=1479
Frame 43 (63 bytes on wire, 63 bytes captured)
Ethernet II, Src: 00:0a:f4:32:b7:ff, Dst: 00:60:08:82:d2:19
Internet Protocol, Src Addr: 202.108.44.206 (202.108.44.206), Dst Addr: 10.214.6.15 (10.214.6.15)
Version: 4
Header length: 20 bytes
Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
0000 00.. = Differentiated Services Codepoint: Default (0x00)
.... ..0. = ECN-Capable Transport (ECT): 0
.... ...0 = ECN-CE: 0
Total Length: 49
Identification: 0x3571 (13681)
Flags: 0x04 (Don't Fragment)
0... = Reserved bit: Not set
.1.. = Don't fragment: Set
..0. = More fragments: Not set
Fragment offset: 0
Time to live: 51
Protocol: TCP (0x06)
Header checksum: 0x0a37 (correct)
Source: 202.108.44.206 (202.108.44.206)
Destination: 10.214.6.15 (10.214.6.15)
Transmission Control Protocol, Src Port: smtp (25), Dst Port: 1686 (1686), Seq: 318, Ack: 1479, Len: 9
Source port: smtp (25)
Destination port: 1686 (1686)
Sequence number: 318 (relative sequence number)
[Next sequence number: 327 (relative sequence number)]
Acknowledgement number: 1479 (relative ack number)
Header length: 20 bytes
Flags: 0x0018 (PSH, ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 1... = Push: Set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
.... ...0 = Fin: Not set
Window size: 8028
Checksum: 0x5e09 (correct)
Simple Mail Transfer Protocol
Response: 221 Bye\r\n
Response code: 221
Response parameter: Bye
0000 00 60 08 82 d2 19 00 0a f4 32 b7 ff 08 00 45 00
0010 00 31 35 71 40 00 33 06 0a 37 ca 6c 2c ce 0a d6
0020 06 0f 00 19 06 96 20 09 7f d9 99 a6 d5 2d 50 18
0030 1f 5c 5e 09 00 00 32 32 31 20 42 79 65 0d 0a
File: mail smtp 5682 bytes 00:00:20
P: 49 D: 49 M: 0

图 9-53

接下来就是熟悉的 TCP 4 次握手了，如图 9-54～图 9-57 所示。

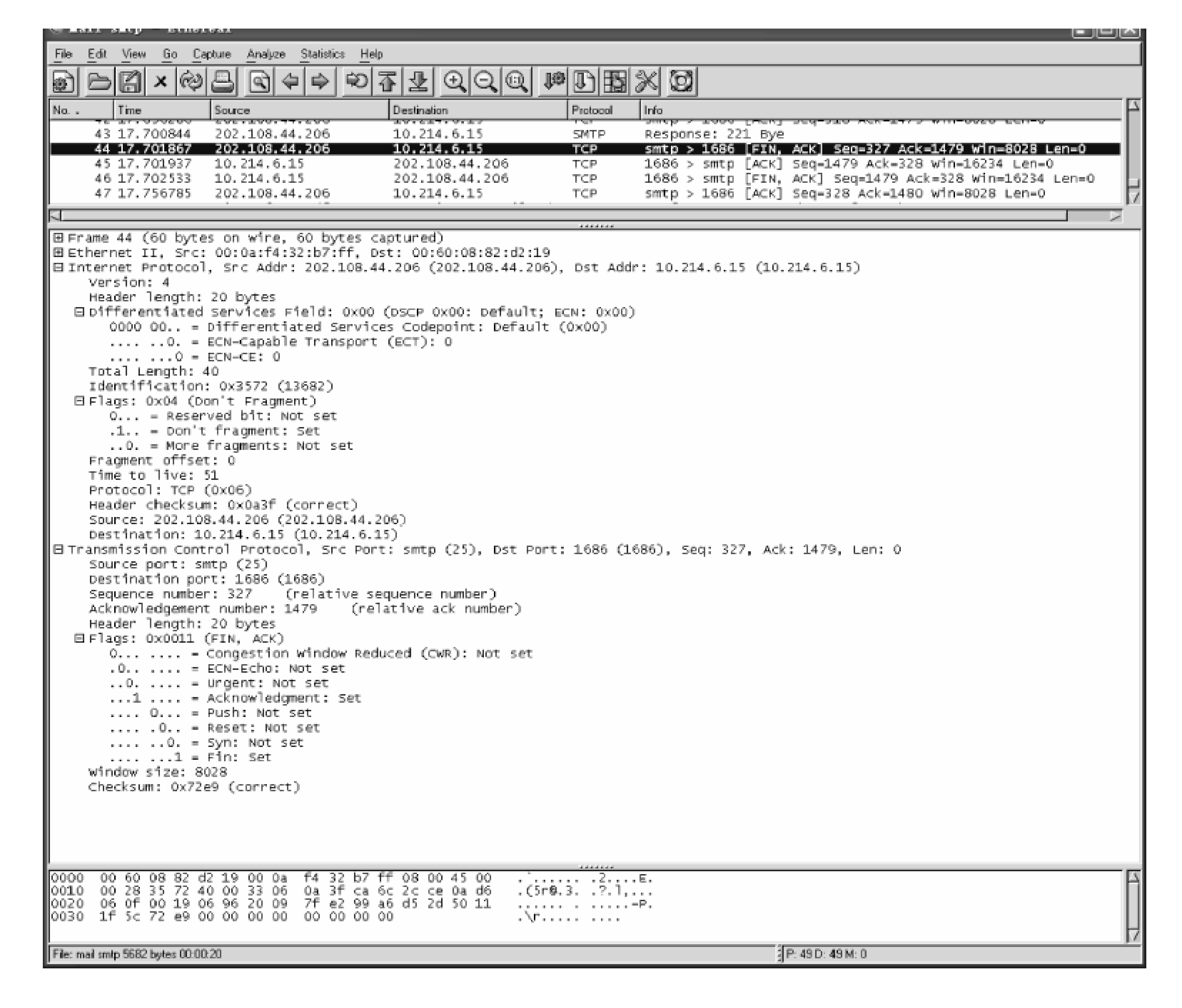


图　9-54

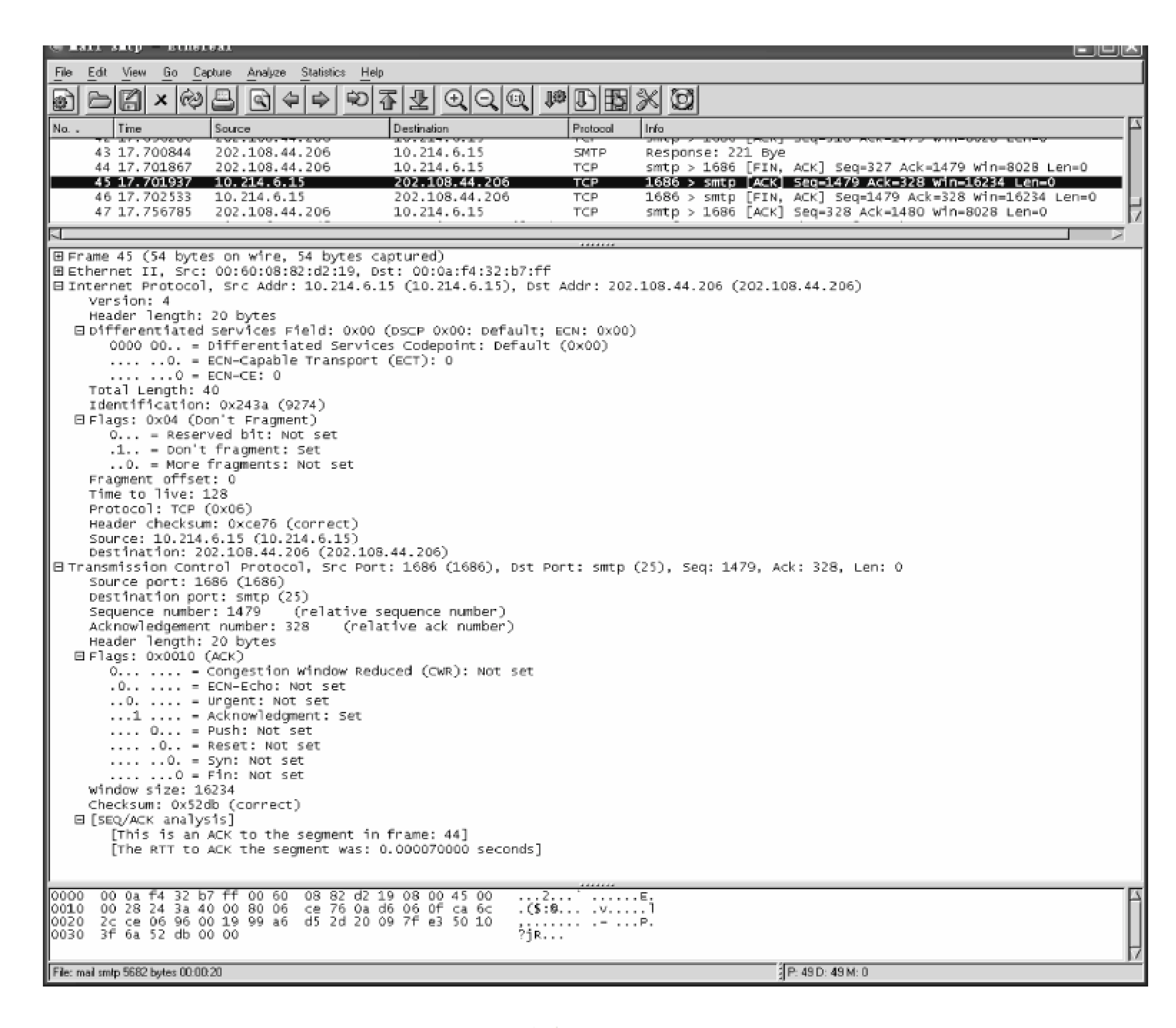


图　9-55

```
File  Edit  View  Go  Capture  Analyze  Statistics  Help

No. .  Time        Source           Destination      Protocol  Info
    43 17.700844   202.108.44.206   10.214.6.15      SMTP      Response: 221 Bye
    44 17.701867   202.108.44.206   10.214.6.15      TCP       smtp > 1686 [FIN, ACK] Seq=327 Ack=1479 Win=8028 Len=0
    45 17.701937   10.214.6.15      202.108.44.206   TCP       1686 > smtp [ACK] Seq=1479 Ack=328 Win=16234 Len=0
    46 17.702533   10.214.6.15      202.108.44.206   TCP       1686 > smtp [FIN, ACK] Seq=1479 Ack=328 Win=16234 Len=0
    47 17.756785   202.108.44.206   10.214.6.15      TCP       smtp > 1686 [ACK] Seq=328 Ack=1480 Win=8028 Len=0

⊞ Frame 46 (54 bytes on wire, 54 bytes captured)
⊞ Ethernet II, Src: 00:60:08:82:d2:19, Dst: 00:0a:f4:32:b7:ff
⊟ Internet Protocol, Src Addr: 10.214.6.15 (10.214.6.15), Dst Addr: 202.108.44.206 (202.108.44.206)
    Version: 4
    Header length: 20 bytes
  ⊟ Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
      0000 00.. = Differentiated Services Codepoint: Default (0x00)
      .... ..0. = ECN-Capable Transport (ECT): 0
      .... ...0 = ECN-CE: 0
    Total Length: 40
    Identification: 0x243b (9275)
  ⊟ Flags: 0x04 (Don't Fragment)
      0... = Reserved bit: Not set
      .1.. = Don't fragment: Set
      ..0. = More fragments: Not set
    Fragment offset: 0
    Time to live: 128
    Protocol: TCP (0x06)
    Header checksum: 0xce75 (correct)
    Source: 10.214.6.15 (10.214.6.15)
    Destination: 202.108.44.206 (202.108.44.206)
⊟ Transmission Control Protocol, Src Port: 1686 (1686), Dst Port: smtp (25), Seq: 1479, Ack: 328, Len: 0
    Source port: 1686 (1686)
    Destination port: smtp (25)
    Sequence number: 1479    (relative sequence number)
    Acknowledgement number: 328    (relative ack number)
    Header length: 20 bytes
  ⊟ Flags: 0x0011 (FIN, ACK)
      0... .... = Congestion Window Reduced (CWR): Not set
      .0.. .... = ECN-Echo: Not set
      ..0. .... = Urgent: Not set
      ...1 .... = Acknowledgment: Set
      .... 0... = Push: Not set
      .... .0.. = Reset: Not set
      .... ..0. = Syn: Not set
      .... ...1 = Fin: Set
    Window size: 16234
    Checksum: 0x52da (correct)

0000  00 0a f4 32 b7 ff 00 60  08 82 d2 19 08 00 45 00   ...2...` ......E.
0010  00 28 24 3b 40 00 80 06  ce 75 0a d6 06 0f ca 6c   .($;@... .u.....l
0020  2c ce 06 96 00 19 99 a6  d5 2d 20 09 7f e3 50 11   ,....... .- ...P.
0030  3f 6a 52 da 00 00                                  ?jR...

File: mail smtp 5682 bytes 00:00:20                P: 49 D: 49 M: 0
```

图 9-56

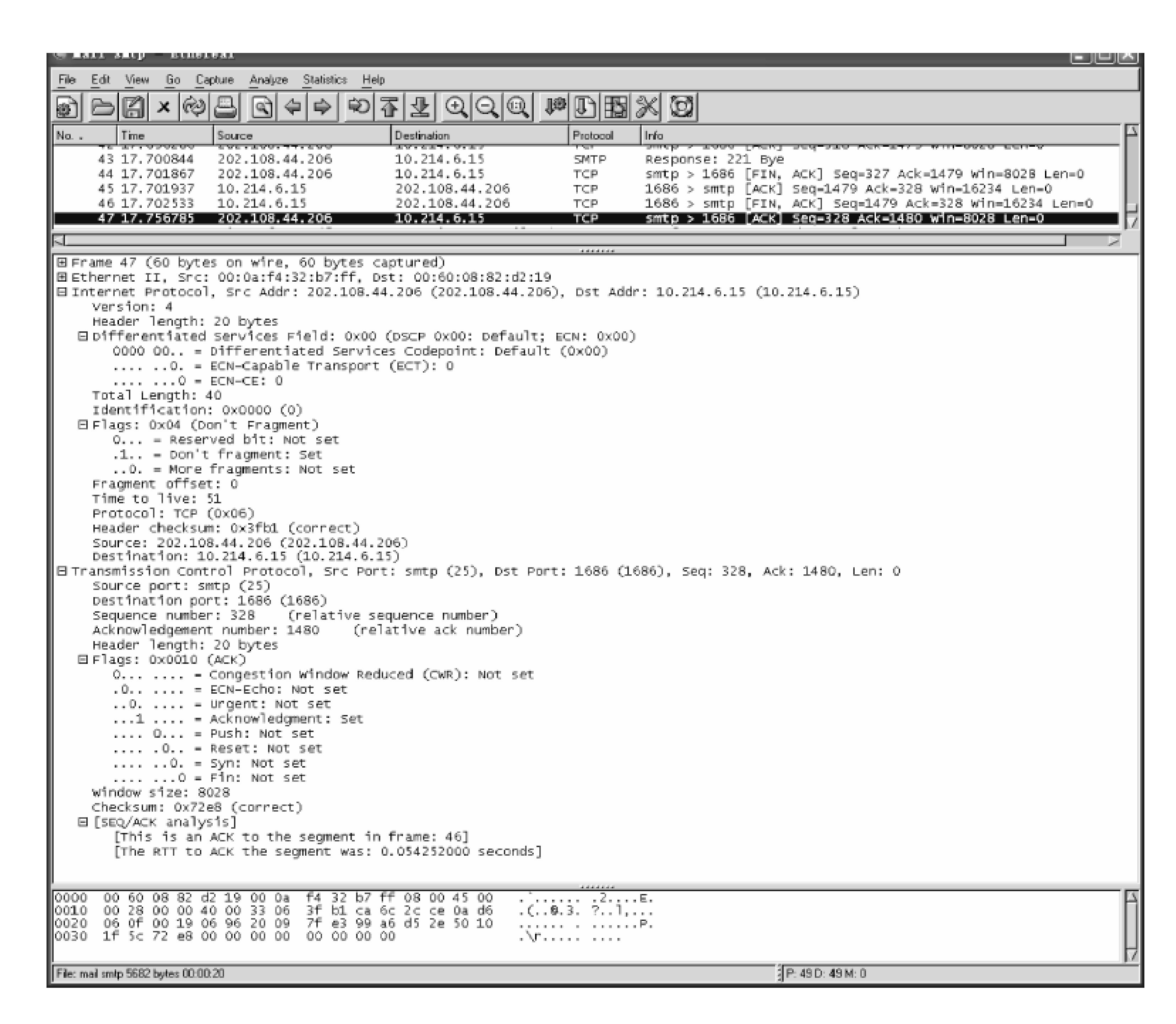



图 9-57

CHAPTER 10

# 第 10 章 DHCP 协议

## 10.1 DHCP 协议概述

DHCP(Dynamic Host Configuration Protocol)分为两个部分：一个是服务器端，另一个是客户端。DHCP 服务器集中管理网络环境资料并负责处理客户端的 DHCP 请求，客户端根据从服务器分配下来的 IP 环境资料来设置自身的网络环境。DHCP 分配地址有两种方式。

(1) 自动分配：一旦 DHCP 客户端如下第一次成功地从 DHCP 服务器端租用到 IP 位址之后，就永远使用这个位址。

(2) 动态分配：当 DHCP 第一次从 DHCP 服务器端租用到 IP 地址之后，并非永久的使用该位址，只要租约到期，客户端就得释放(release)这个 IP 位址，以便其他主机可以使用。但是客户端有权比其他主机更优先地延续(renew)租约，或是租用其他的 IP 位址。由此可见动态分配显然比自动分配更加灵活，尤其是当 IP 地址址不够用的时候。

DHCP 的前身是 BOOTP，BOOTP 协议是一个基于 TCP/IP 协议的协议，它可以让无盘工作站从一个中心服务器上获得 IP 地址，这样为局域网中的无盘工作站分配动态 IP 地址，并不需要每个用户去设置静态 IP 地址。DHCP 是对 BOOTP 的扩展，它的包格式和 BOOTP 一样，这使得 BOOTP 和 DHCP 之间可以实现互操作。它们都使用 UDP 端口号是 67(服务器端)和 68(客户端)。DHCP 和 BOOTP 的区别如下：

(1) DHCP 对 IP 地址使用有个时间的限制，在客户使用期限到了就要释放这个地址，或者申请续租这个 IP 地址。

(2) DHCP 包长度比 BOOTP 包长，因此 DHCP 为用户提供所有 IP 网络配置参数。

(3) DHCP 有 7 种消息类型，而 BOOTP 只有两种。

DHCP 既有优点也有缺点。它的优点是：网络管理员可以通过 DHCP 服务器来验证 IP 地址和其他配置参数，而不用去检查每个主机；DHCP 服务

器不会同时租借相同的 IP 地址给两台主机，这样避免了地址冲突；DHCP 管理员可以约束特定的计算机使用特定的 IP 地址；可以为每个 DHCP 作用域设置很多选项；客户机在不同子网间移动时不需要重新设置 IP 地址。

同时它的缺点也存在：DHCP 不能发现网络上非 DHCP 客户机已经在使用的 IP 地址；除非路由器允许 BOOTP 转发，一般 DHCP 服务器不能跨路由器与客户机通信。

DHCP 的工作过程如下：

(1) DHCP 客户机寻找 DHCP 服务器的阶段。如果客户机第一次上网，没有设定 IP 地址等网络信息，用 DHCP 来发现网络初始设置时，DHCP 客户机以广播方式(因为 DHCP 服务器的 IP 地址对于客户机来说是未知的)发送 DHCP discover 报文来寻找 DHCP 服务器，这个广播报文向地址 255.255.255.255 发送。网络上每一台安装了 TCP/IP 协议的主机都会接收到这个广播信息，但只有 DHCP 服务器才会做出响应，如图 10-1 所示。

当客户端将第一个 DHCP discover 报文送出去之后在 1 秒之内没有得到回应的话就会进行第 2 次发送 DHCP discover 报文；第 2 次等待的时间为 9 秒，9 秒后没得到回应，则会第 3 次发送 DHCP discover 报文；第 3 次等待的时间为 13 秒，如果 13 秒后没有回应，就发送第 4 次 DHCP discover 广播；第 4 次等待时间为 16 秒后，16 秒后还没得到回应就息宣告 DHCP discover 的失败，没有找到 DHCP 服务器。之后，经过 5 分钟再重一次 DHCP discover 的要求。由于客户端在开始的时候还没有 IP 地址所以在它发送的 DHCP discover 报文内会带有其 MAC 地址信息并且有一个 XID 编号，用 DHCP 服务器返回应答用。

(2) DHCP 服务器提供 IP 地址的阶段。在网络中接收到 DHCP discover 发现信息的 DHCP 服务器都会做出响应，它从 IP 地址池中挑选一个未分配的 IP 地址给 DHCP 客户端，通过向 DHCP 客户端发送一个包含出租的 IP 地址和其他设置的 DHCP offer 来提供信息如图 10-2 所示。DHCP 服务器回应的 DHCP offer 报文，这个报文含有必要的网络设置信息，诸如 IP 地址、租约期限和网关等信息。

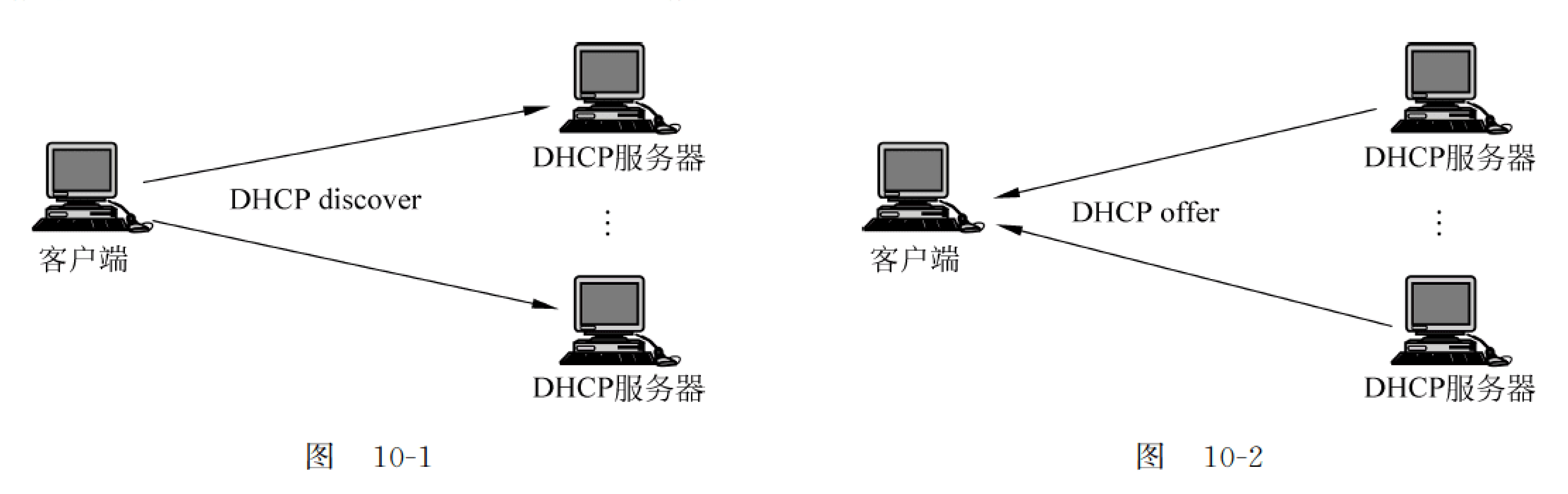



图 10-1　　　　图 10-2

(3) DHCP 客户端选择某台 DHCP 服务器提供的 IP 地址的阶段。如果有多台 DHCP 服务器向 DHCP 客户端发来的 DHCP offer 提供信息，那么 DHCP 客户端只接受第一个收到 DHCP offer 提供的信息，然后它就以广播方式回答一个 DHCP request 请求信息，该信息中包含向它所选定的 DHCP 服务器请求 IP 地址等内容，发送这个请求信息是为了让所有的 DHCP 服务器知道，它将选择某台 DHCP 服务器所提供的 IP 地址如图 10-3 所示，因此，这个请求信息使用的是广播地址。

(4) DHCP 服务器确认所提供的 IP 地址的阶段。当 DHCP 服务器收到 DHCP 客户端

的 DHCP request 请求信息之后，它便向 DHCP 客户端发送一个包含它所提供的 IP 地址和其他设置的 DHCP ack 确认信息，告诉 DHCP 客户端可以使用它所提供的 IP 地址。然后 DHCP 客户端便将其 TCP/IP 协议与网卡绑定，另外，除 DHCP 客户端选中的服务器外，其他的 DHCP 服务器都将收回曾提供的 IP 地址，如图 10-4 所示。

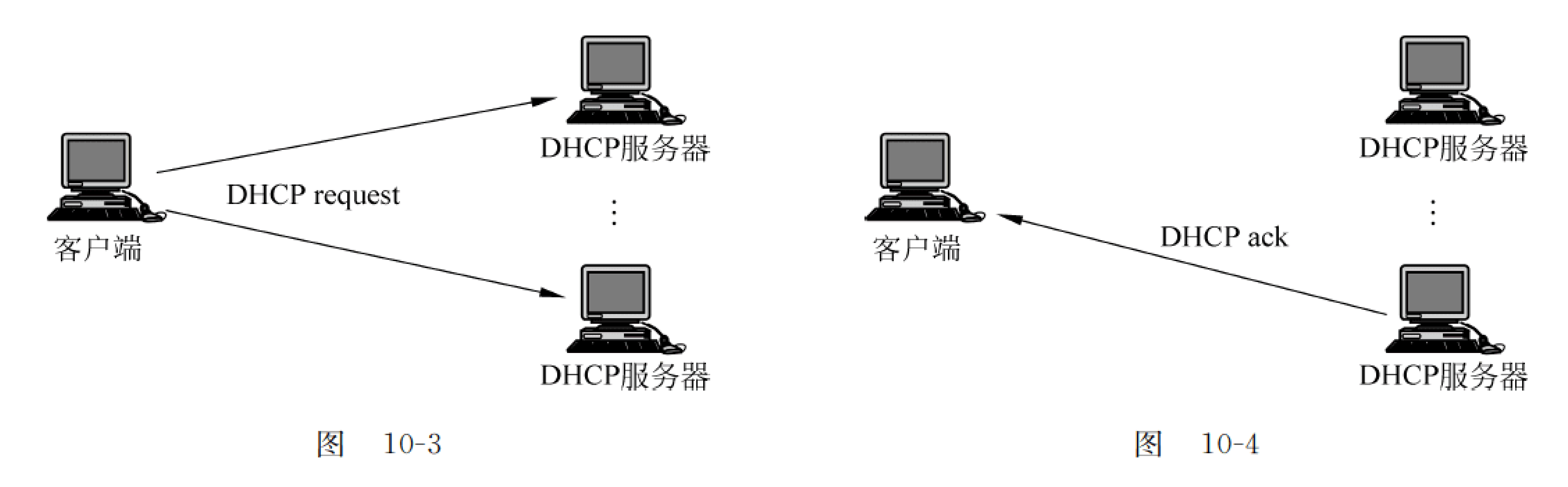



图　10-3　　　　图　10-4

当客户端启用 DHCP 服务器给的 IP 地址前，客户端还会向网络发送一个 ARP(Address Resolution Protocol)报文查询网络上面有没有其他机器使用该 IP 地址，如果发现该 IP 已经被占用，客户端则会送出一个 DHCP decline 报文给 DHCP 服务器拒绝接受其 DHCP offer 并重新发送 DHCP discover 信息。

(5) DHCP 客户端再次重新登录网络时，就不需要再发送 DHCP discover 发现信息了，而是直接发送包含前一次所分配的 IP 地址的 DHCP request 请求信息。当 DHCP 服务器收到这一信息后，它会尝试让 DHCP 客户端继续使用原来的 IP 地址，并回答一个 DHCP ack 确认信息。如果此 IP 地址已无法再分配给原来的 DHCP 客户端使用时(比如此 IP 地址已分配给其他 DHCP 客户端使用)，则 DHCP 服务器给 DHCP 客户端回答一个 DHCP nack 否认信息。当原来的 DHCP 客户端收到此 DHCP nack 否认信息后，它就必须重新发送 DHCP discover 发现信息来请求新的 IP 地址。

(6) DHCP 服务器向 DHCP 客户端出租的 IP 地址一般都有一个租借期限，期满后 DHCP 服务器便会收回出租的 IP 地址。如果 DHCP 客户端继续使用该 IP 地址，则必须更新其 IP 租约。DHCP 客户端在 IP 租约期限过一半时，会自动向 DHCP 服务器发送 DHCP request，更新其 IP 租约的信息。如果此时得不到 DHCP 服务器的确认的话客户端还可以继续使用该 IP，然后在剩下的租约期限的再一半的时候(即租约的 75%)还得不到确认的话，那么工作站就不能拥有这个 IP 了。

## 10.2　DHCP 报文结构

DHCP 报文结构如图 10-5 所示。

1) Opcode

当为 Boot request 时，Opcode 值为 1，当为 Boot reply 时，Opcode 的值为 2。

2) HTYPE

硬件类别，Ethernet 为 1，一些常见的硬件类别及值如表 10-1 所示。

| 0 | 8 | 16 | 24 | 31 |
|---|---|---|---|---|
| Opcode | HTYPE | HLEN | HOPS | |
| Transaction ID | | | | |
| Seconds | | Flags | | |
| Client IP address | | | | |
| Your IP address | | | | |
| Server IP address | | | | |
| Gateway IP address | | | | |
| Client hardware address | | | | |
| Server host name | | | | |
| Boot filename | | | | |
| Options | | | | |

图　10-5

**表 10-1　常见的硬件类别及值**

| 值 | 描　　述 |
|---|---|
| 1 | Ethernet |
| 3 | Amateur Radio AX. 25. |
| 4 | Proteon ProNET Token Ring |
| 5 | Chaos |
| 6 | IEEE 802 |
| 7 | ARCNET |
| 8 | Hyperchannel |
| 9 | Lanstar |
| 10 | Autonet Short Address |
| 11 | LocalTalk |
| 12 | LocalNet (IBM PCNet or SYTEK LocalNET) |
| 13 | Ultra link |
| 14 | SMDS |
| 15 | Frame Relay |
| 16 | ATM，Asynchronous Transmission Mode |
| 17 | HDLC |
| 18 | Fibre Channel |
| 19 | ATM，Asynchronous Transmission Mode |
| 20 | Serial Line |
| 27 | EUI-64 |
| 31 | IPsec tunnel |

3）HLEN

硬件地址长度，Ethernet 为 6。

4）HOPS

定义报文可经过的最大跳数。在同一网内为 0。

5）Transacion ID

客户端发送 DHCP request 时，选择一个随机的数字，以作为 DHCP 服务器端回应时的依据。

6）Seconds

客户端启动时间(秒)，当客户端获得一个地址或者更新进程。

7）Flages

0～15 共 16 位，最高位为 1 时表示服务器将以广播方式传送报文给客户，其余尚未使用。

8）Clinet IP address

客户端想继续使用以前取得的 IP 地址，这里填写想继续使用的 IP 地址。

9）Your IP address

这里填写分配给客户的 IP 位址。

10）Server IP address

如果客户需要通过网络启动，这里填写启动程序所在的服务器的地址。

11）Gateway IP address

如果需跨网段进行 DHCP 发放，这个字段为中继代理的地址，否则为 0。

12）Client hareware address

客户端的硬件地址。

13）Server host name

服务器的名称，以 0x00 结尾。

14）Boot filename

如果客户端需要通过网络开机，这里将填写引导文件的名称，稍后以 TFTP 传送。

15）Options

提供更多的附加信息(如 Netmask、Gateway、DNS 等)或者厂商信息，其长度可变。

16）Magic cookie

这个字段出现表明下一个数据段就是选项。

## 10.3　DHCP 报文分析

接下来看看实际截获的 DHCP 的报文：

如图 10-6 所示，客户端发送一个 DHCP discover 报文，这个报文的源地址为 0.0.0.0，目标地址为广播地址 255.255.255.255。DHCP discover 报文是个 Boot request 类型的报文，注意这个报文的 Transaction ID，后续的报文都是凭借这个 ID 来维持服务器和客户端联系的。报文中还包含了客户端的 MAC 地址、主机名等信息。

如图 10-7 所示，服务器端收到 DHCP discover 报文后，发送 DHCP offer 报文，Source 地址是 DHCP 服务器 IP 地址，Destination 地址是 255.255.255.255，可以看到报文中有一个分给客户端的 IP 地址，DHCP Option Field 部分，可以看到由服务器端随 IP 地址一起发送子网掩码、默认网关(路由器)、租约时间和域名服务器地址。

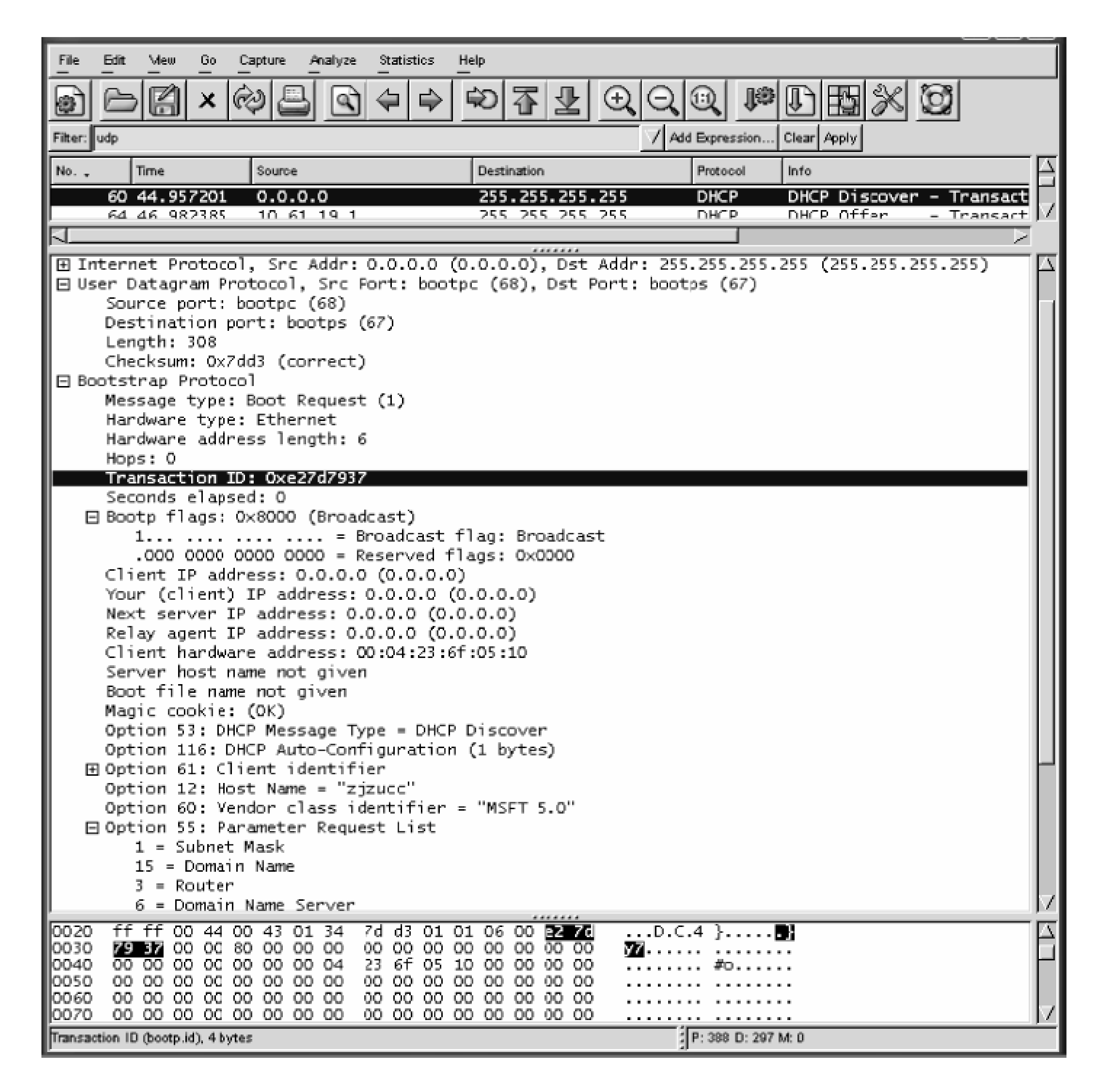

File Edit View Go Capture Analyze Statistics Help
Filter: udp
Add Expression... Clear Apply
No. Time Source Destination Protocol Info
60 44.957201 0.0.0.0 255.255.255.255 DHCP DHCP Discover - Transact
64 46.982385 10.61.19.1 255.255.255.255 DHCP DHCP Offer - Transact
Internet Protocol, Src Addr: 0.0.0.0 (0.0.0.0), Dst Addr: 255.255.255.255 (255.255.255.255)
User Datagram Protocol, Src Port: bootpc (68), Dst Port: bootps (67)
Source port: bootpc (68)
Destination port: bootps (67)
Length: 308
Checksum: 0x7dd3 (correct)
Bootstrap Protocol
Message type: Boot Request (1)
Hardware type: Ethernet
Hardware address length: 6
Hops: 0
Transaction ID: 0xe27d7937
Seconds elapsed: 0
Bootp flags: 0x8000 (Broadcast)
1... .... .... .... = Broadcast flag: Broadcast
.000 0000 0000 0000 = Reserved flags: 0x0000
Client IP address: 0.0.0.0 (0.0.0.0)
Your (client) IP address: 0.0.0.0 (0.0.0.0)
Next server IP address: 0.0.0.0 (0.0.0.0)
Relay agent IP address: 0.0.0.0 (0.0.0.0)
Client hardware address: 00:04:23:6f:05:10
Server host name not given
Boot file name not given
Magic cookie: (OK)
Option 53: DHCP Message Type = DHCP Discover
Option 116: DHCP Auto-Configuration (1 bytes)
Option 61: Client identifier
Option 12: Host Name = "zjzucc"
Option 60: Vendor class identifier = "MSFT 5.0"
Option 55: Parameter Request List
1 = Subnet Mask
15 = Domain Name
3 = Router
6 = Domain Name Server
0020 ff ff 00 44 00 43 01 34 7d d3 01 01 06 00 e2 7d ...D.C.4 }......}
0030 79 37 00 00 80 00 00 00 00 00 00 00 00 00 00 00 y7...... ........
0040 00 00 00 00 00 00 00 04 23 6f 05 10 00 00 00 00 ........ #o......
0050 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ........ ........
0060 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ........ ........
0070 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ........ ........
Transaction ID (bootp.id), 4 bytes
P: 388 D: 297 M: 0

图 10-6

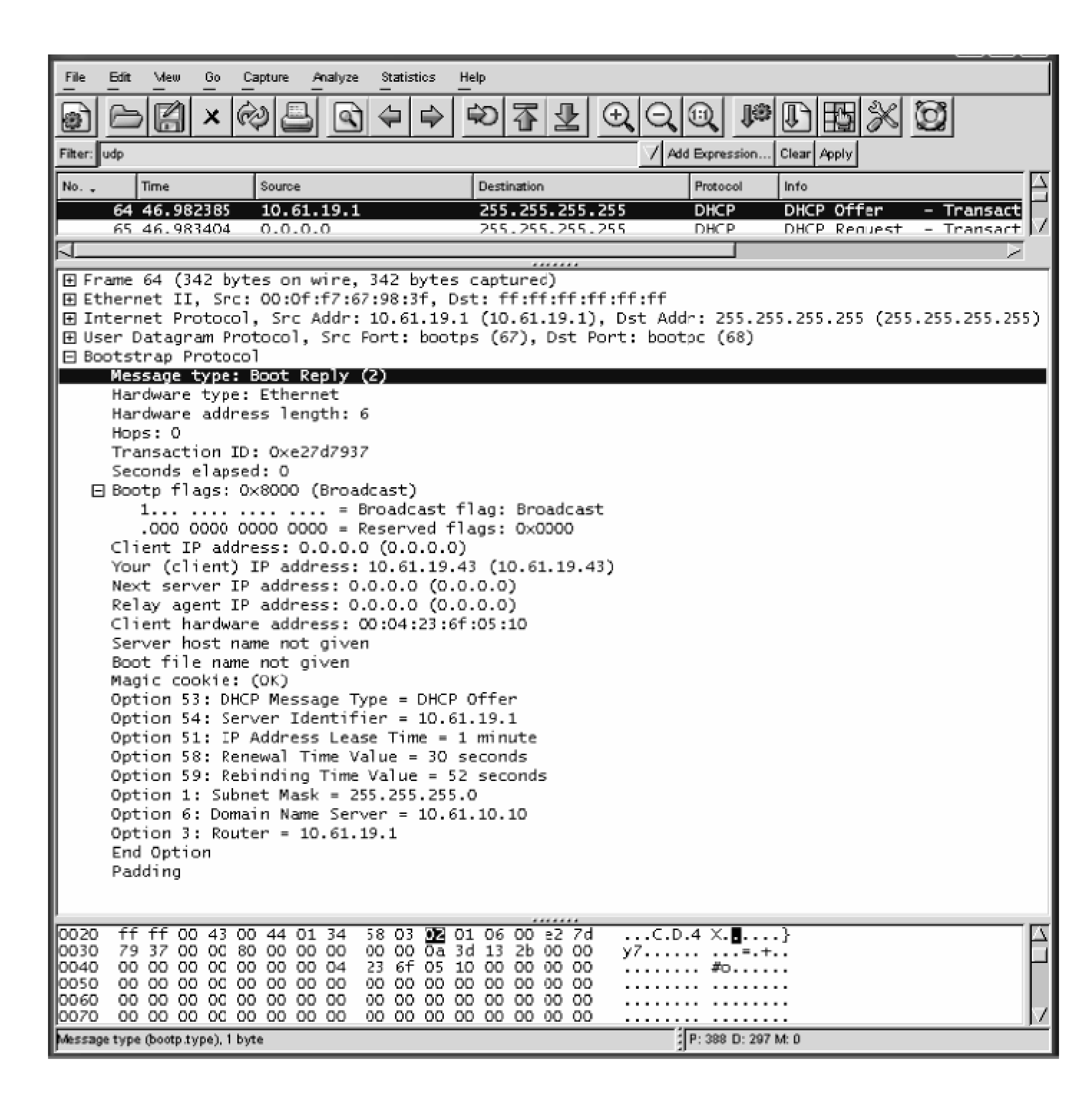

File Edit View Go Capture Analyze Statistics Help
Filter: udp
Add Expression... Clear Apply
No. Time Source Destination Protocol Info
64 46.982385 10.61.19.1 255.255.255.255 DHCP DHCP Offer - Transact
65 46.983404 0.0.0.0 255.255.255.255 DHCP DHCP Request - Transact
Frame 64 (342 bytes on wire, 342 bytes captured)
Ethernet II, Src: 00:0f:f7:67:98:3f, Dst: ff:ff:ff:ff:ff:ff
Internet Protocol, Src Addr: 10.61.19.1 (10.61.19.1), Dst Addr: 255.255.255.255 (255.255.255.255)
User Datagram Protocol, Src Port: bootps (67), Dst Port: bootpc (68)
Bootstrap Protocol
Message type: Boot Reply (2)
Hardware type: Ethernet
Hardware address length: 6
Hops: 0
Transaction ID: 0xe27d7937
Seconds elapsed: 0
Bootp flags: 0x8000 (Broadcast)
1... .... .... .... = Broadcast flag: Broadcast
.000 0000 0000 0000 = Reserved flags: 0x0000
Client IP address: 0.0.0.0 (0.0.0.0)
Your (client) IP address: 10.61.19.43 (10.61.19.43)
Next server IP address: 0.0.0.0 (0.0.0.0)
Relay agent IP address: 0.0.0.0 (0.0.0.0)
Client hardware address: 00:04:23:6f:05:10
Server host name not given
Boot file name not given
Magic cookie: (OK)
Option 53: DHCP Message Type = DHCP Offer
Option 54: Server Identifier = 10.61.19.1
Option 51: IP Address Lease Time = 1 minute
Option 58: Renewal Time Value = 30 seconds
Option 59: Rebinding Time Value = 52 seconds
Option 1: Subnet Mask = 255.255.255.0
Option 6: Domain Name Server = 10.61.10.10
Option 3: Router = 10.61.19.1
End Option
Padding
0020 ff ff 00 43 00 44 01 34 58 03 02 01 06 00 e2 7d ...C.D.4 X.......}
0030 79 37 00 00 80 00 00 00 00 00 0a 3d 13 2b 00 00 y7...... ...=.+..
0040 00 00 00 00 00 00 00 04 23 6f 05 10 00 00 00 00 ........ #o......
0050 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ........ ........
0060 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ........ ........
0070 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ........ ........
Message type (bootp.type), 1 byte
P: 388 D: 297 M: 0

图 10-7

如图 10-8 所示，客户端收到 DHCP offer 报文后，打算使用服务器端分配的 IP 地址，这时，客户端发送 DHCP request，使用 Requested IP Address 字段以向服务器确认客户端将使用这个地址，Server Identifier 字段显示提供租约的 DHCP 服务器的 IP 地址。

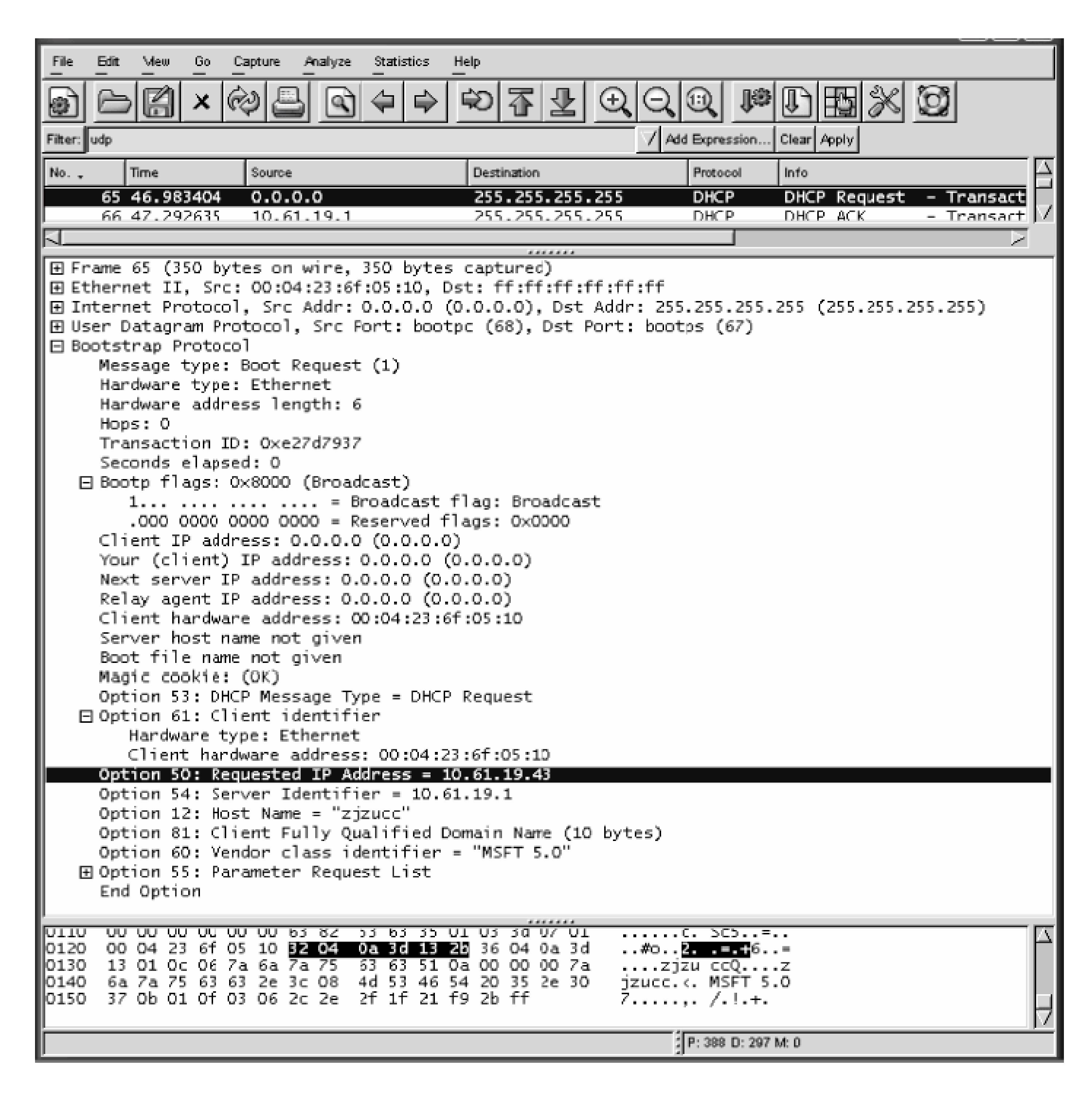


图 10-8

如图 10-9 所示，DHCP 服务器用 DHCP ack 响应 DHCP request，以此完成初始化周期。Source 地址是 DHCP 服务器的 IP 地址，Destination 地址仍然是 255.255.255.255。YIADDR 字段包含客户端的地址，而 Client MAC address 字段是发出请求的客户端中网卡的物理地址。DHCP Option 部分将数据包标识为 ACK。

如图 10-10 所示，客户端启用地址前，发送 ARP 请求，检测是否网络中已有主机使用了从 DHCP 服务器端分配的 IP 地址，如果没有主机响应这个 ARP 请求，则正式启用这个 IP 地址。

如图 10-11 所示，由于从 DHCP 服务器上得来的 IP 地址有个租用时间，期满后 DHCP 服务器便会收回出租的 IP 地址。如果要继续使用该地址，就要更新租约，DHCP 客户端在 IP 租约期限过一半时，会自动向 DHCP 服务器发送 DHCP request，更新 IP 租约的信息。

如图 10-12 所示，服务器同意客户端继续使用该 IP 地址。

接下来看看，初始时设置过 IP 地址 10.61.19.2，然后启用 DHCP 来获得 IP 地址时的报文情况。

如图 10-13 所示，这是个 DHCP discover 报文，可以看到这个报文中 option 中有个字段 requested IP address＝10.61.19.2，这个字段是客户端请求 DHCP 服务器分配客户端曾经使用过的 10.61.19.2 这个地址给客户端。

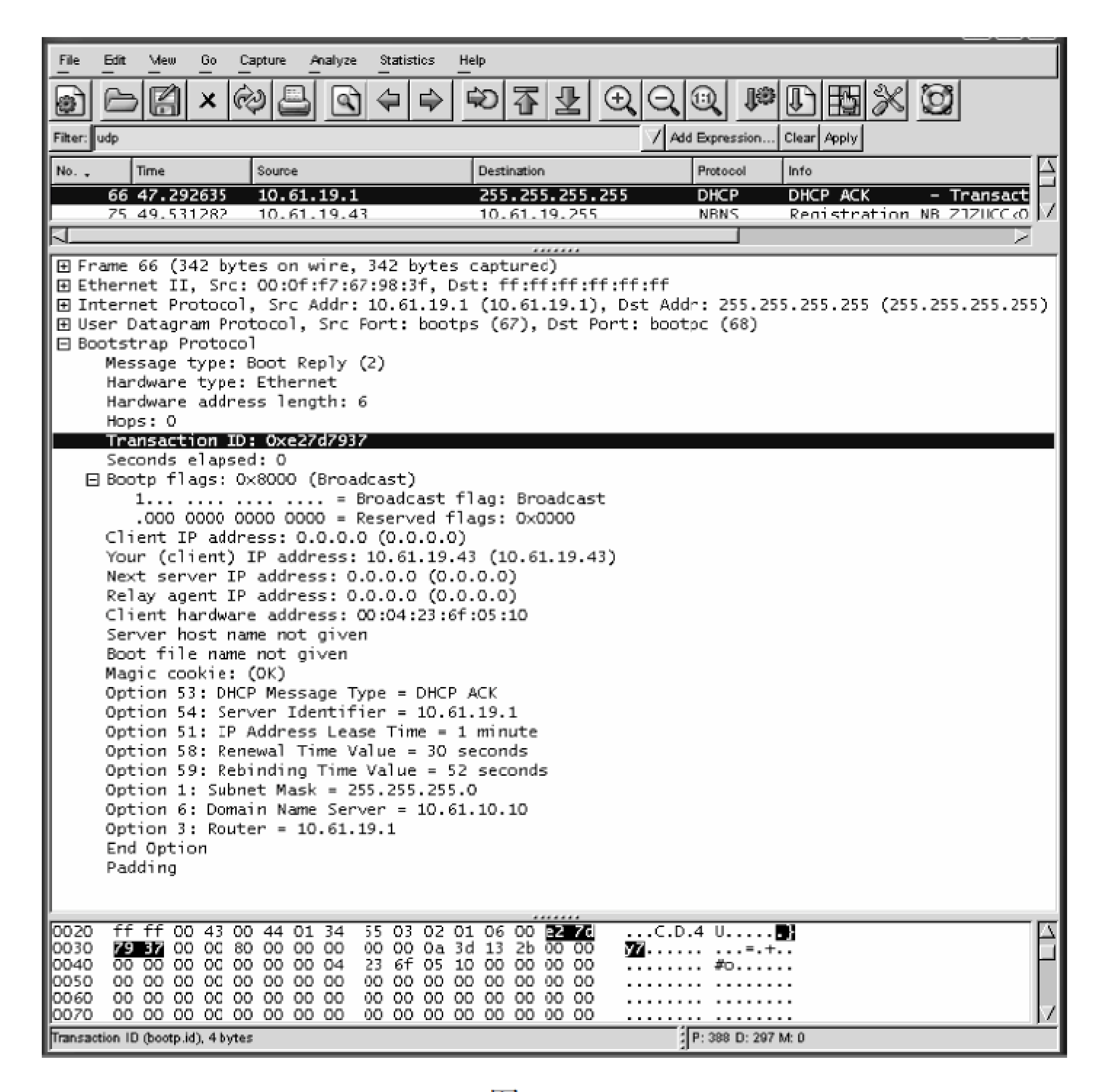



图 10-9

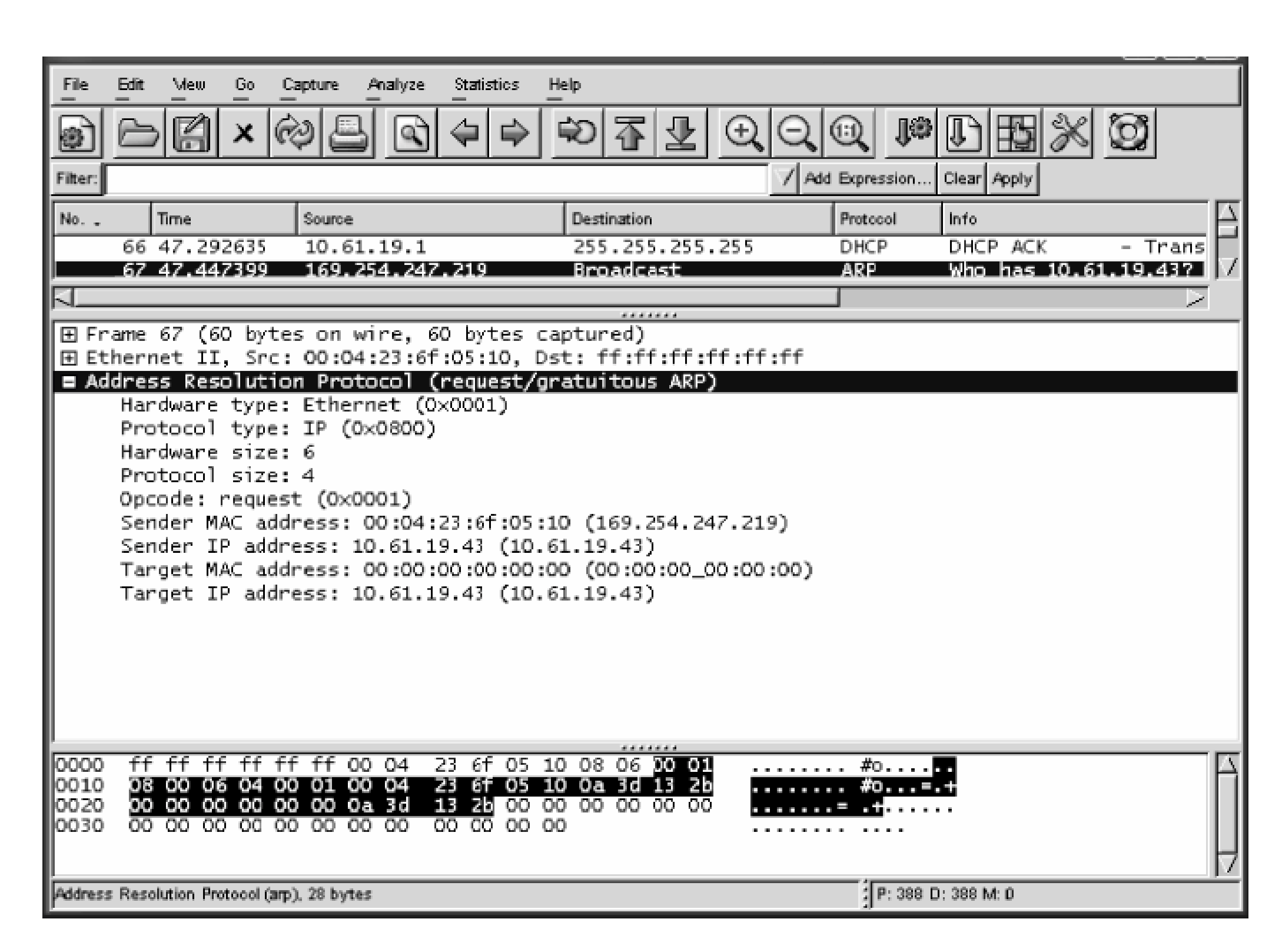



图 10-10

如图 10-14 所示，服务器端分配给客户端一个新的 IP 地址 10.61.19.151。如图 10-15 所示，客户端确认将使用新的 IP 地址 10.61.19.151。

如图 10-16 所示，DHCP 服务器用 DHCP ack 响应 DHCP request，以此完成初始化周期。

如图 10-17 所示，随后，客户端发送 ARP 请求，以确认 10.61.19.151 是否有人使用。如果没收到回应，则正式使用 10.61.19.151。

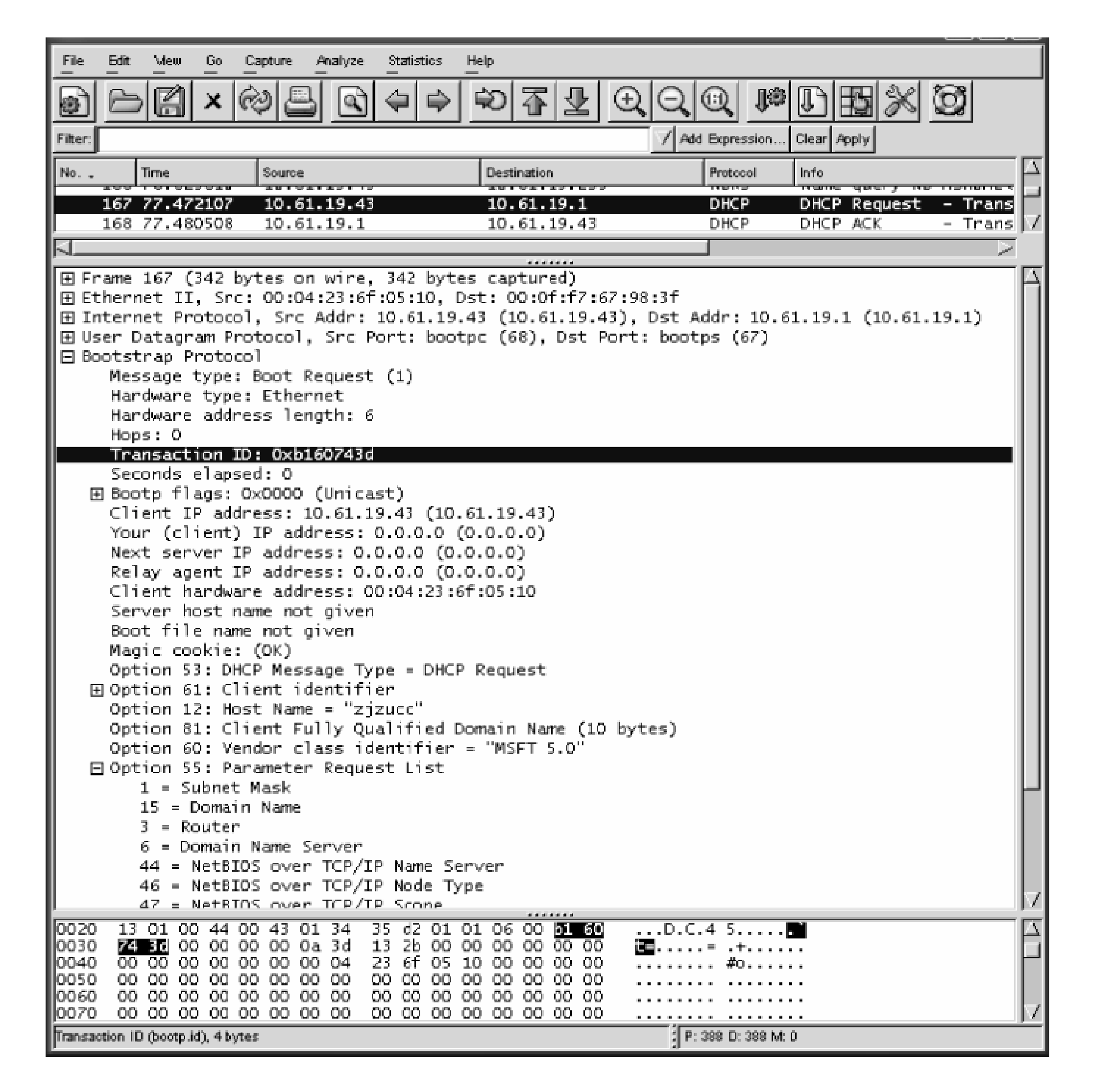

File Edit View Go Capture Analyze Statistics Help
Filter:
Add Expression... Clear Apply
No. Time Source Destination Protocol Info
167 77.472107 10.61.19.43 10.61.19.1 DHCP DHCP Request - Trans
168 77.480508 10.61.19.1 10.61.19.43 DHCP DHCP ACK - Trans
Frame 167 (342 bytes on wire, 342 bytes captured)
Ethernet II, Src: 00:04:23:6f:05:10, Dst: 00:0f:f7:67:98:3f
Internet Protocol, Src Addr: 10.61.19.43 (10.61.19.43), Dst Addr: 10.61.19.1 (10.61.19.1)
User Datagram Protocol, Src Port: bootpc (68), Dst Port: bootps (67)
Bootstrap Protocol
Message type: Boot Request (1)
Hardware type: Ethernet
Hardware address length: 6
Hops: 0
Transaction ID: 0xb160743d
Seconds elapsed: 0
Bootp flags: 0x0000 (Unicast)
Client IP address: 10.61.19.43 (10.61.19.43)
Your (client) IP address: 0.0.0.0 (0.0.0.0)
Next server IP address: 0.0.0.0 (0.0.0.0)
Relay agent IP address: 0.0.0.0 (0.0.0.0)
Client hardware address: 00:04:23:6f:05:10
Server host name not given
Boot file name not given
Magic cookie: (OK)
Option 53: DHCP Message Type = DHCP Request
Option 61: Client identifier
Option 12: Host Name = "zjzucc"
Option 81: Client Fully Qualified Domain Name (10 bytes)
Option 60: Vendor class identifier = "MSFT 5.0"
Option 55: Parameter Request List
1 = Subnet Mask
15 = Domain Name
3 = Router
6 = Domain Name Server
44 = NetBIOS over TCP/IP Name Server
46 = NetBIOS over TCP/IP Node Type
0020 13 01 00 44 00 43 01 34 35 c2 01 01 06 00 b1 60 ...D.C.4 5.....
0030 74 3d 00 00 00 00 0a 3d 13 2b 00 00 00 00 00 00 .......= .+......
0040 00 00 00 00 00 00 00 04 23 6f 05 10 00 00 00 00 ........ #o......
0050 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ........ ........
0060 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ........ ........
0070 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ........ ........
Transaction ID (bootp.id), 4 bytes
P: 388 D: 388 M: 0

图　10-11

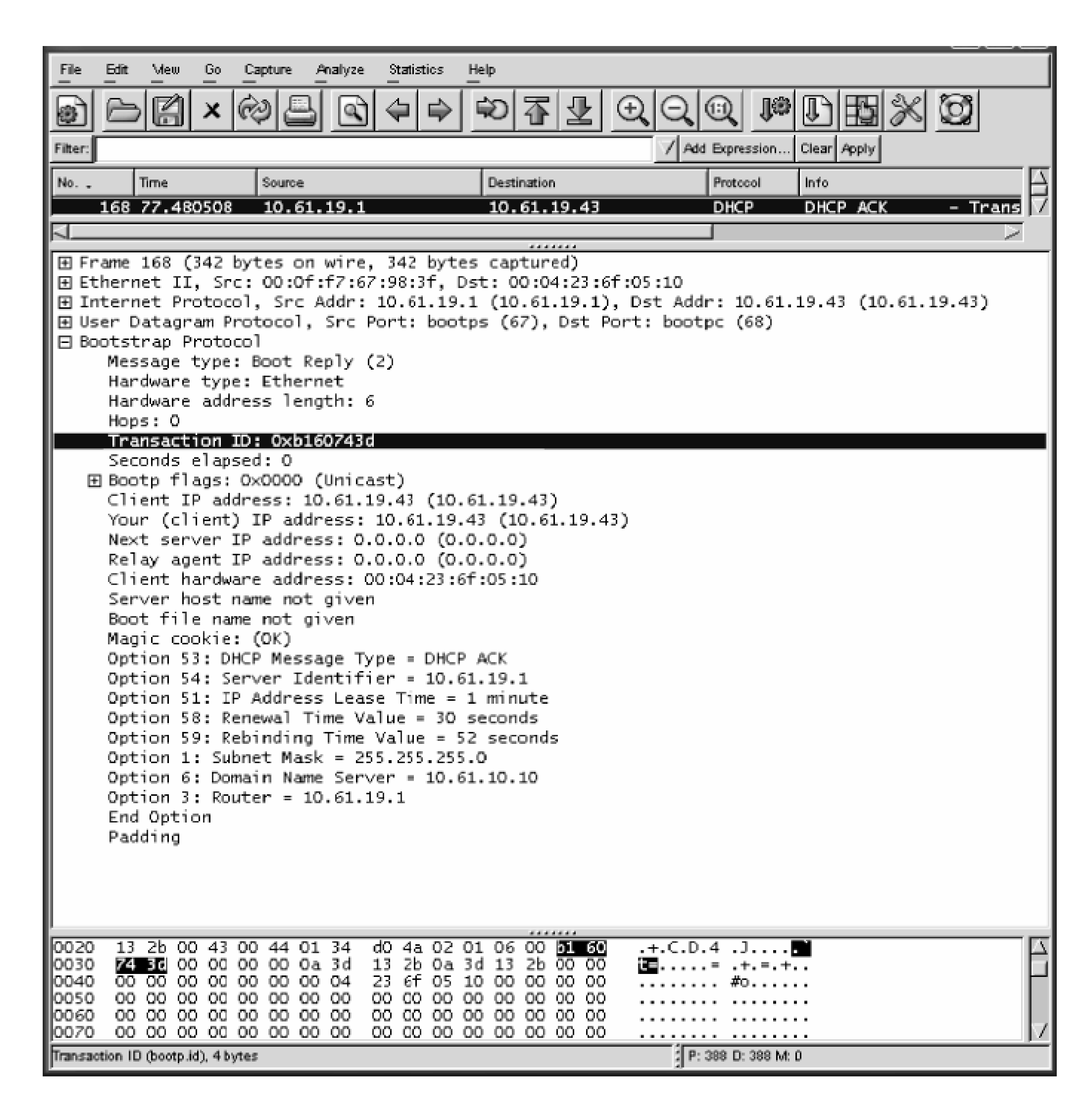

File Edit View Go Capture Analyze Statistics Help
Filter:
Add Expression... Clear Apply
No. Time Source Destination Protocol Info
168 77.480508 10.61.19.1 10.61.19.43 DHCP DHCP ACK - Trans
Frame 168 (342 bytes on wire, 342 bytes captured)
Ethernet II, Src: 00:0f:f7:67:98:3f, Dst: 00:04:23:6f:05:10
Internet Protocol, Src Addr: 10.61.19.1 (10.61.19.1), Dst Addr: 10.61.19.43 (10.61.19.43)
User Datagram Protocol, Src Port: bootps (67), Dst Port: bootpc (68)
Bootstrap Protocol
Message type: Boot Reply (2)
Hardware type: Ethernet
Hardware address length: 6
Hops: 0
Transaction ID: 0xb160743d
Seconds elapsed: 0
Bootp flags: 0x0000 (Unicast)
Client IP address: 10.61.19.43 (10.61.19.43)
Your (client) IP address: 10.61.19.43 (10.61.19.43)
Next server IP address: 0.0.0.0 (0.0.0.0)
Relay agent IP address: 0.0.0.0 (0.0.0.0)
Client hardware address: 00:04:23:6f:05:10
Server host name not given
Boot file name not given
Magic cookie: (OK)
Option 53: DHCP Message Type = DHCP ACK
Option 54: Server Identifier = 10.61.19.1
Option 51: IP Address Lease Time = 1 minute
Option 58: Renewal Time Value = 30 seconds
Option 59: Rebinding Time Value = 52 seconds
Option 1: Subnet Mask = 255.255.255.0
Option 6: Domain Name Server = 10.61.10.10
Option 3: Router = 10.61.19.1
End Option
Padding
0020 13 2b 00 43 00 44 01 34 d0 4a 02 01 06 00 b1 60 .+.C.D.4 .J......
0030 74 3d 00 00 00 00 0a 3d 13 2b 0a 3d 13 2b 00 00 .......= .+.=.+..
0040 00 00 00 00 00 00 00 04 23 6f 05 10 00 00 00 00 ........ #o......
0050 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ........ ........
0060 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ........ ........
0070 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ........ ........
Transaction ID (bootp.id), 4 bytes
P: 388 D: 388 M: 0

图　10-12



图 10-13



图 10-14

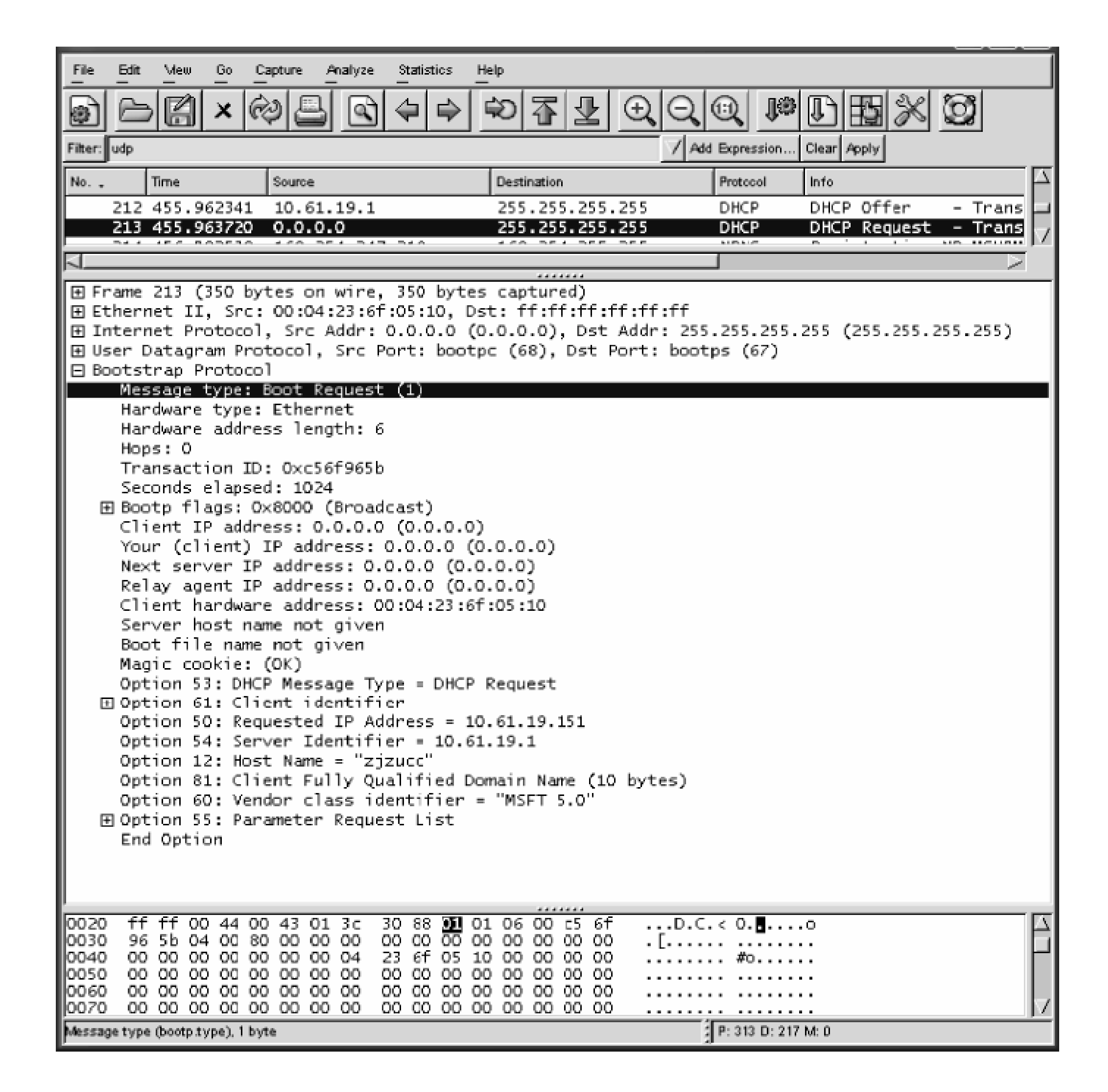

File Edit View Go Capture Analyze Statistics Help
Filter: udp
Add Expression... Clear Apply
No. Time Source Destination Protocol Info
212 455.962341 10.61.19.1 255.255.255.255 DHCP DHCP Offer - Trans
213 455.963720 0.0.0.0 255.255.255.255 DHCP DHCP Request - Trans
Frame 213 (350 bytes on wire, 350 bytes captured)
Ethernet II, Src: 00:04:23:6f:05:10, Dst: ff:ff:ff:ff:ff:ff
Internet Protocol, Src Addr: 0.0.0.0 (0.0.0.0), Dst Addr: 255.255.255.255 (255.255.255.255)
User Datagram Protocol, Src Port: bootpc (68), Dst Port: bootps (67)
Bootstrap Protocol
Message type: Boot Request (1)
Hardware type: Ethernet
Hardware address length: 6
Hops: 0
Transaction ID: 0xc56f965b
Seconds elapsed: 1024
Bootp flags: 0x8000 (Broadcast)
Client IP address: 0.0.0.0 (0.0.0.0)
Your (client) IP address: 0.0.0.0 (0.0.0.0)
Next server IP address: 0.0.0.0 (0.0.0.0)
Relay agent IP address: 0.0.0.0 (0.0.0.0)
Client hardware address: 00:04:23:6f:05:10
Server host name not given
Boot file name not given
Magic cookie: (OK)
Option 53: DHCP Message Type = DHCP Request
Option 61: Client identifier
Option 50: Requested IP Address = 10.61.19.151
Option 54: Server Identifier = 10.61.19.1
Option 12: Host Name = "zjzucc"
Option 81: Client Fully Qualified Domain Name (10 bytes)
Option 60: Vendor class identifier = "MSFT 5.0"
Option 55: Parameter Request List
End Option
0020 ff ff 00 44 00 43 01 3c 30 88 01 01 06 00 c5 6f
0030 96 5b 04 00 80 00 00 00 00 00 00 00 00 00 00 00
0040 00 00 00 00 00 00 00 04 23 6f 05 10 00 00 00 00
0050 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0060 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0070 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
Message type (bootp.type), 1 byte
P: 313 D: 217 M: 0

图　10-15

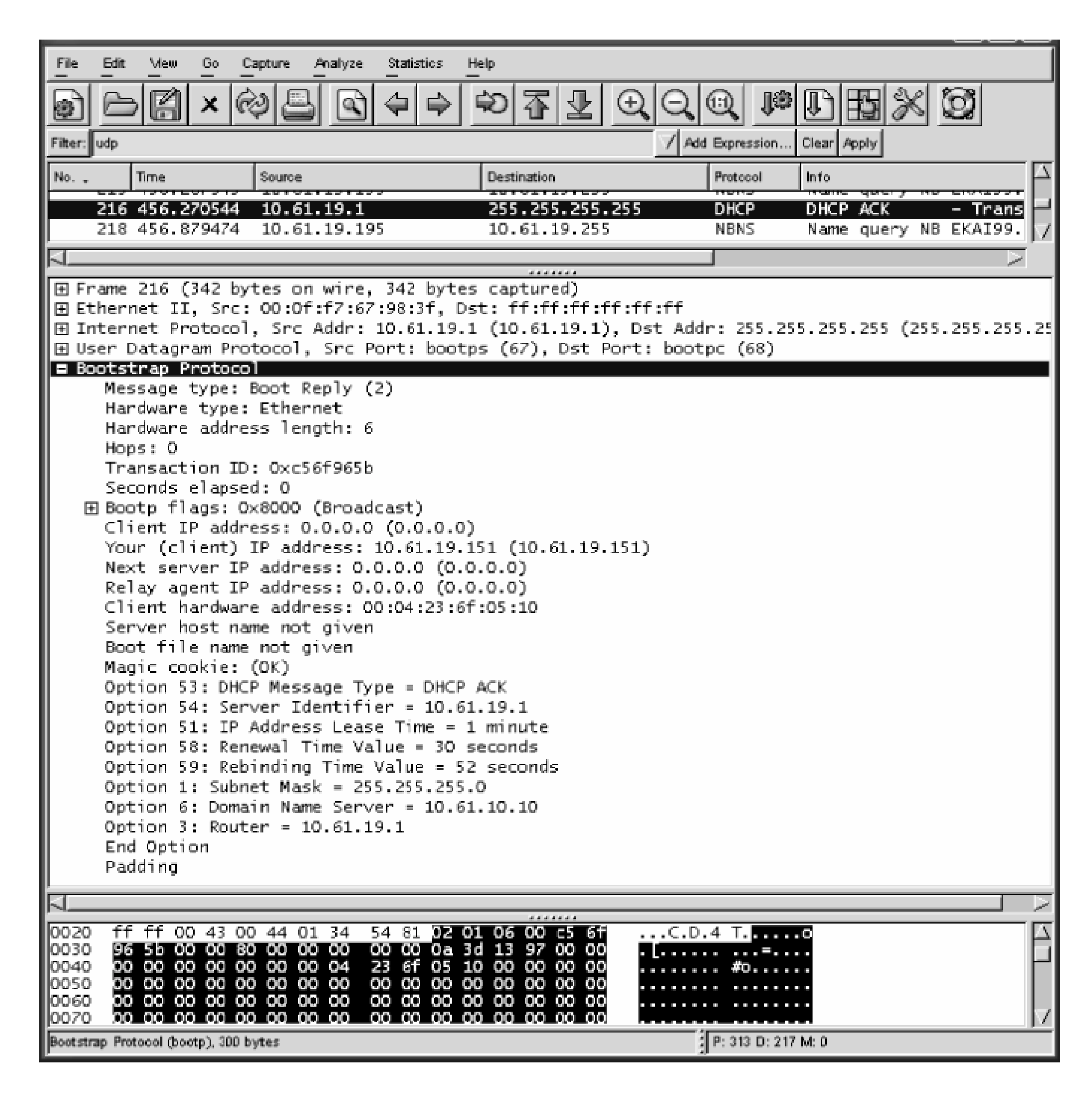

File Edit View Go Capture Analyze Statistics Help
Filter: udp
Add Expression... Clear Apply
No. Time Source Destination Protocol Info
216 456.270544 10.61.19.1 255.255.255.255 DHCP DHCP ACK - Trans
218 456.879474 10.61.19.195 10.61.19.255 NBNS Name query NB EKAI99.
Frame 216 (342 bytes on wire, 342 bytes captured)
Ethernet II, Src: 00:0f:f7:67:98:3f, Dst: ff:ff:ff:ff:ff:ff
Internet Protocol, Src Addr: 10.61.19.1 (10.61.19.1), Dst Addr: 255.255.255.255 (255.255.255.25
User Datagram Protocol, Src Port: bootps (67), Dst Port: bootpc (68)
Bootstrap Protocol
Message type: Boot Reply (2)
Hardware type: Ethernet
Hardware address length: 6
Hops: 0
Transaction ID: 0xc56f965b
Seconds elapsed: 0
Bootp flags: 0x8000 (Broadcast)
Client IP address: 0.0.0.0 (0.0.0.0)
Your (client) IP address: 10.61.19.151 (10.61.19.151)
Next server IP address: 0.0.0.0 (0.0.0.0)
Relay agent IP address: 0.0.0.0 (0.0.0.0)
Client hardware address: 00:04:23:6f:05:10
Server host name not given
Boot file name not given
Magic cookie: (OK)
Option 53: DHCP Message Type = DHCP ACK
Option 54: Server Identifier = 10.61.19.1
Option 51: IP Address Lease Time = 1 minute
Option 58: Renewal Time Value = 30 seconds
Option 59: Rebinding Time Value = 52 seconds
Option 1: Subnet Mask = 255.255.255.0
Option 6: Domain Name Server = 10.61.10.10
Option 3: Router = 10.61.19.1
End Option
Padding
0020 ff ff 00 43 00 44 01 34 54 81 02 01 06 00 c5 6f
0030 96 5b 00 00 80 00 00 00 00 00 0a 3d 13 97 00 00
0040 00 00 00 00 00 00 00 04 23 6f 05 10 00 00 00 00
0050 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0060 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0070 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
Bootstrap Protocol (bootp), 300 bytes
P: 313 D: 217 M: 0

图　10-16

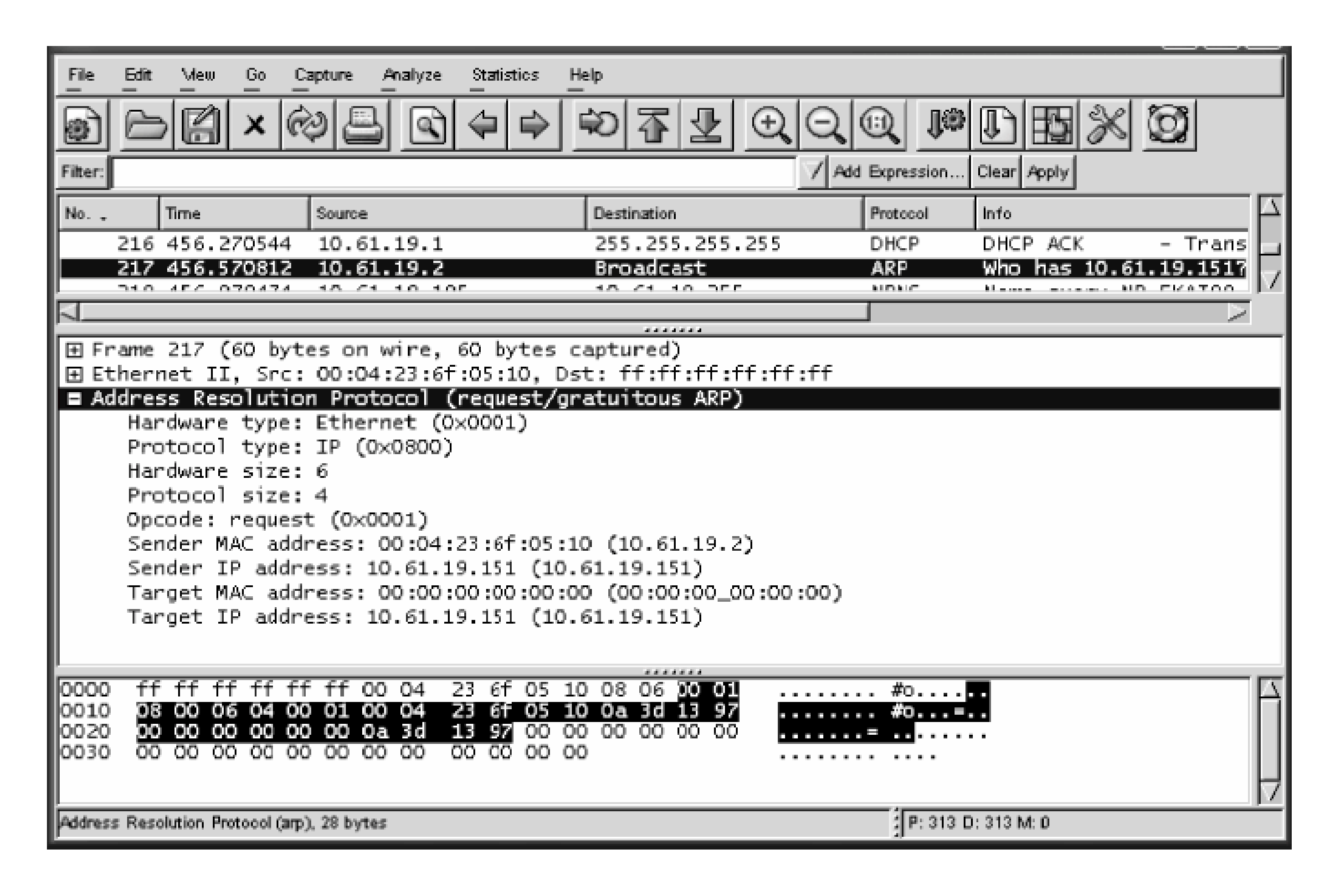



图 10-17

如果服务器端分配给客户端的 IP 地址,客户端经过 ARP 地址检测,发现已经被使用了,如图 10-18 所示,源 MAC 地址 00:13:02:97:4a:83 发送 ARP 地址广播,询问 10.61.19.177 这个地址是否有人使用。如图 10-19 所示,源地址 00:13:02:2a:fa:53 向目标地址 00:13:02:97:4a:83 发送了 ARP 回应,告诉 00:13:02:97:4a:83,IP 地址 10.61.19.177 正被 00:13:02:2a:fa:53 使用。随后,客户端将发送 DHCP decline 报文,如图 10-20 所示,00:13:02:97:4a:83 发送了一个 DHCP decline 广播,绝接受服务器 DHCP offer 的 IP 地址 10.61.19.177。

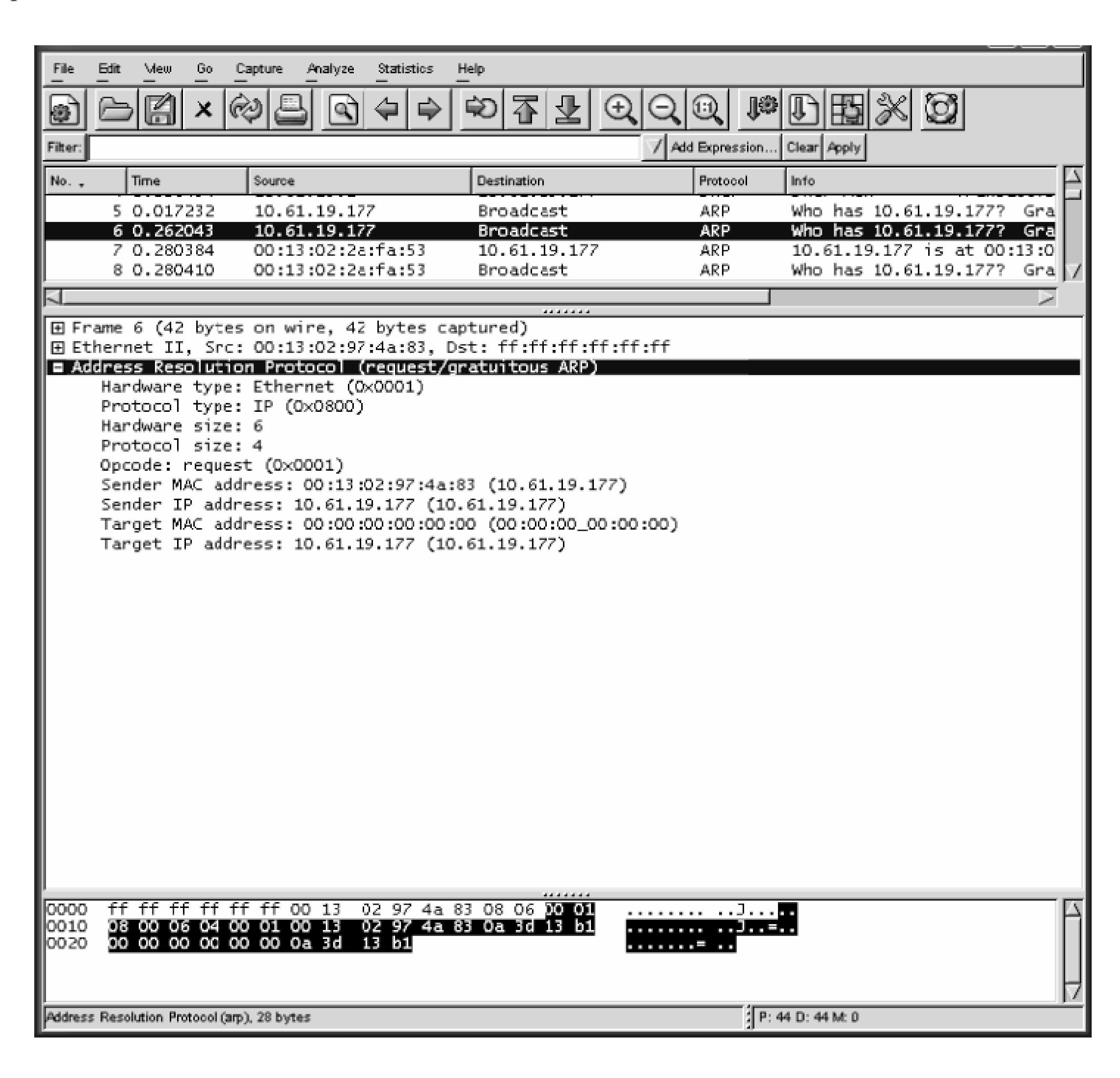



图 10-18

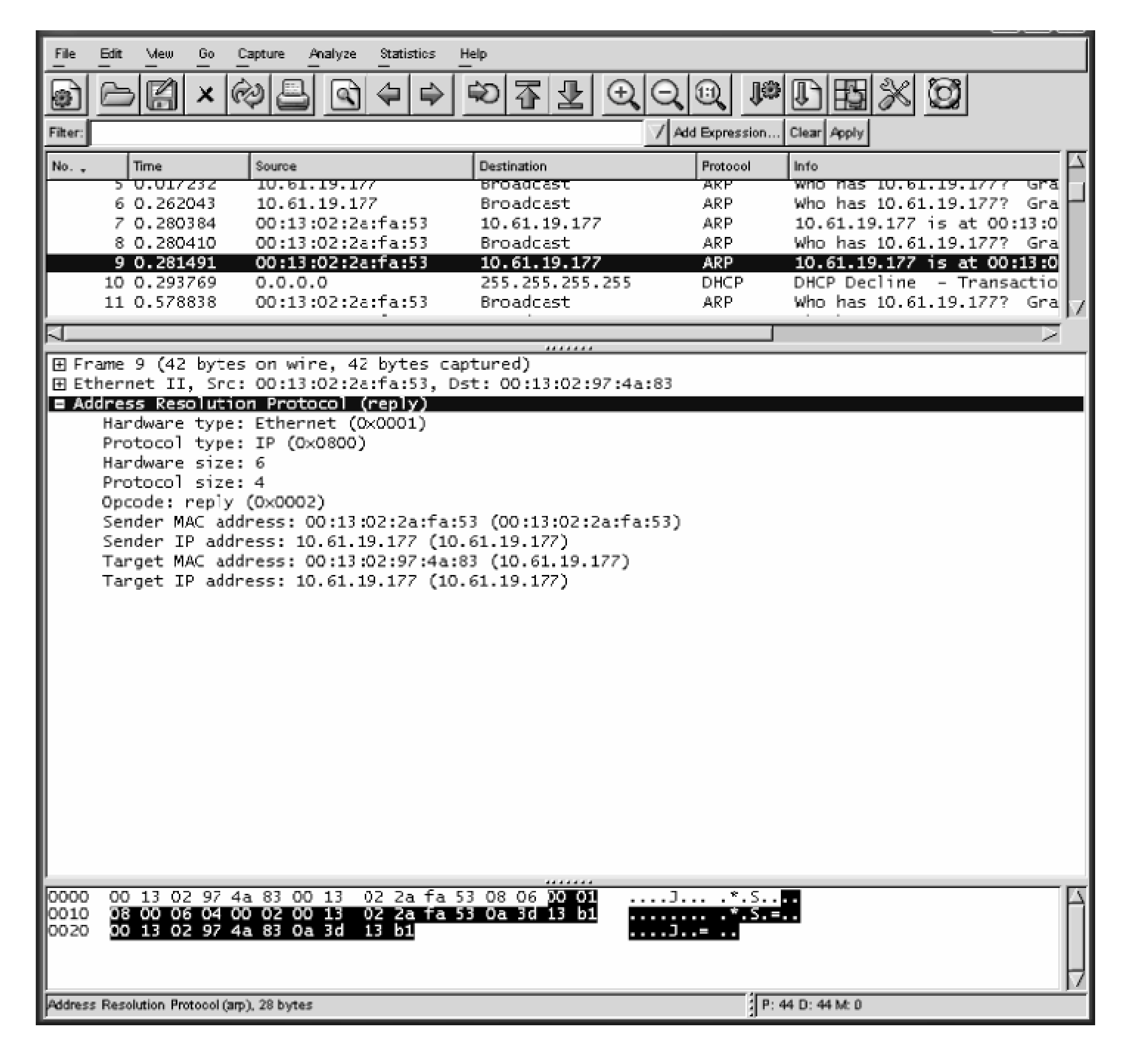

File Edit View Go Capture Analyze Statistics Help
Filter: Add Expression... Clear Apply
No. Time Source Destination Protocol Info
6 0.262043 10.61.19.177 Broadcast ARP Who has 10.61.19.177? Gra
7 0.280384 00:13:02:2a:fa:53 10.61.19.177 ARP 10.61.19.177 is at 00:13:0
8 0.280410 00:13:02:2a:fa:53 Broadcast ARP Who has 10.61.19.177? Gra
9 0.281491 00:13:02:2a:fa:53 10.61.19.177 ARP 10.61.19.177 is at 00:13:0
10 0.293769 0.0.0.0 255.255.255.255 DHCP DHCP Decline - Transactio
11 0.578838 00:13:02:2a:fa:53 Broadcast ARP Who has 10.61.19.177? Gra
Frame 9 (42 bytes on wire, 42 bytes captured)
Ethernet II, Src: 00:13:02:2a:fa:53, Dst: 00:13:02:97:4a:83
Address Resolution Protocol (reply)
Hardware type: Ethernet (0x0001)
Protocol type: IP (0x0800)
Hardware size: 6
Protocol size: 4
Opcode: reply (0x0002)
Sender MAC address: 00:13:02:2a:fa:53 (00:13:02:2a:fa:53)
Sender IP address: 10.61.19.177 (10.61.19.177)
Target MAC address: 00:13:02:97:4a:83 (10.61.19.177)
Target IP address: 10.61.19.177 (10.61.19.177)
0000 00 13 02 97 4a 83 00 13 02 2a fa 53 08 06 00 01
0010 08 00 06 04 00 02 00 13 02 2a fa 53 0a 3d 13 b1
0020 00 13 02 97 4a 83 0a 3d 13 b1
Address Resolution Protocol (arp), 28 bytes
P: 44 D: 44 M: 0

图　10-19

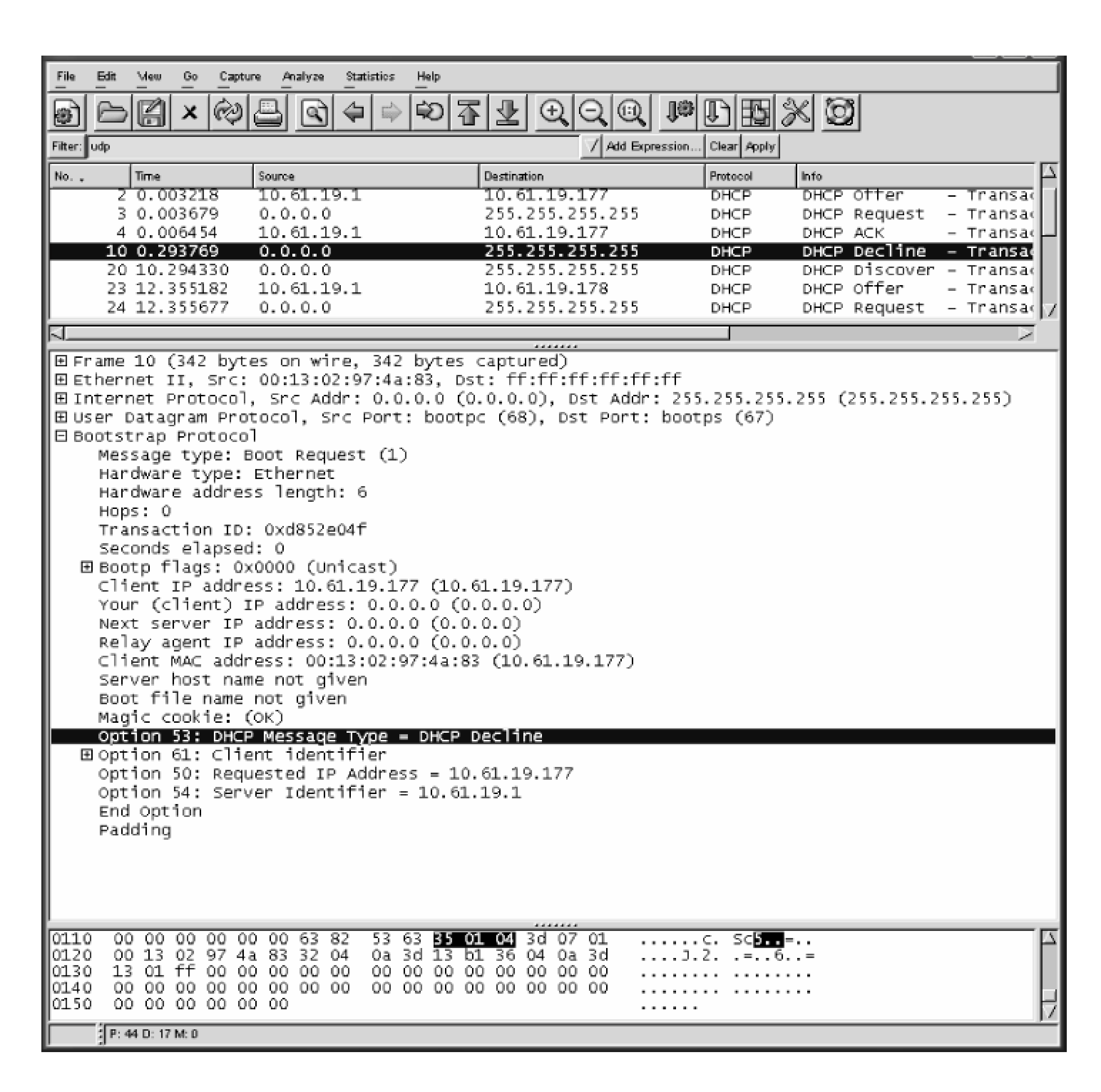

File Edit View Go Capture Analyze Statistics Help
Filter: udp Add Expression... Clear Apply
No. Time Source Destination Protocol Info
3 0.003679 0.0.0.0 255.255.255.255 DHCP DHCP Request - Transa
4 0.006454 10.61.19.1 10.61.19.177 DHCP DHCP ACK - Transa
10 0.293769 0.0.0.0 255.255.255.255 DHCP DHCP Decline - Transa
20 10.294330 0.0.0.0 255.255.255.255 DHCP DHCP Discover - Transa
23 12.355182 10.61.19.1 10.61.19.178 DHCP DHCP Offer - Transa
24 12.355677 0.0.0.0 255.255.255.255 DHCP DHCP Request - Transa
Frame 10 (342 bytes on wire, 342 bytes captured)
Ethernet II, Src: 00:13:02:97:4a:83, Dst: ff:ff:ff:ff:ff:ff
Internet Protocol, Src Addr: 0.0.0.0 (0.0.0.0), Dst Addr: 255.255.255.255 (255.255.255.255)
User Datagram Protocol, Src Port: bootpc (68), Dst Port: bootps (67)
Bootstrap Protocol
Message type: Boot Request (1)
Hardware type: Ethernet
Hardware address length: 6
Hops: 0
Transaction ID: 0xd852e04f
Seconds elapsed: 0
Bootp flags: 0x0000 (Unicast)
Client IP address: 10.61.19.177 (10.61.19.177)
Your (client) IP address: 0.0.0.0 (0.0.0.0)
Next server IP address: 0.0.0.0 (0.0.0.0)
Relay agent IP address: 0.0.0.0 (0.0.0.0)
Client MAC address: 00:13:02:97:4a:83 (10.61.19.177)
Server host name not given
Boot file name not given
Magic cookie: (OK)
Option 53: DHCP Message Type = DHCP Decline
Option 61: Client identifier
Option 50: Requested IP Address = 10.61.19.177
Option 54: Server Identifier = 10.61.19.1
End Option
Padding
0110 00 00 00 00 00 00 63 82 53 63 35 01 04 3d 07 01
0120 00 13 02 97 4a 83 32 04 0a 3d 13 b1 36 04 0a 3d
0130 13 01 ff 00 00 00 00 00 00 00 00 00 00 00 00 00
0140 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0150 00 00 00 00 00 00
P: 44 D: 17 M: 0

图　10-20

随后,客户端发送 DHCP discover 开始一个新的申请 IP 地址的流程,如图 10-21 所示。

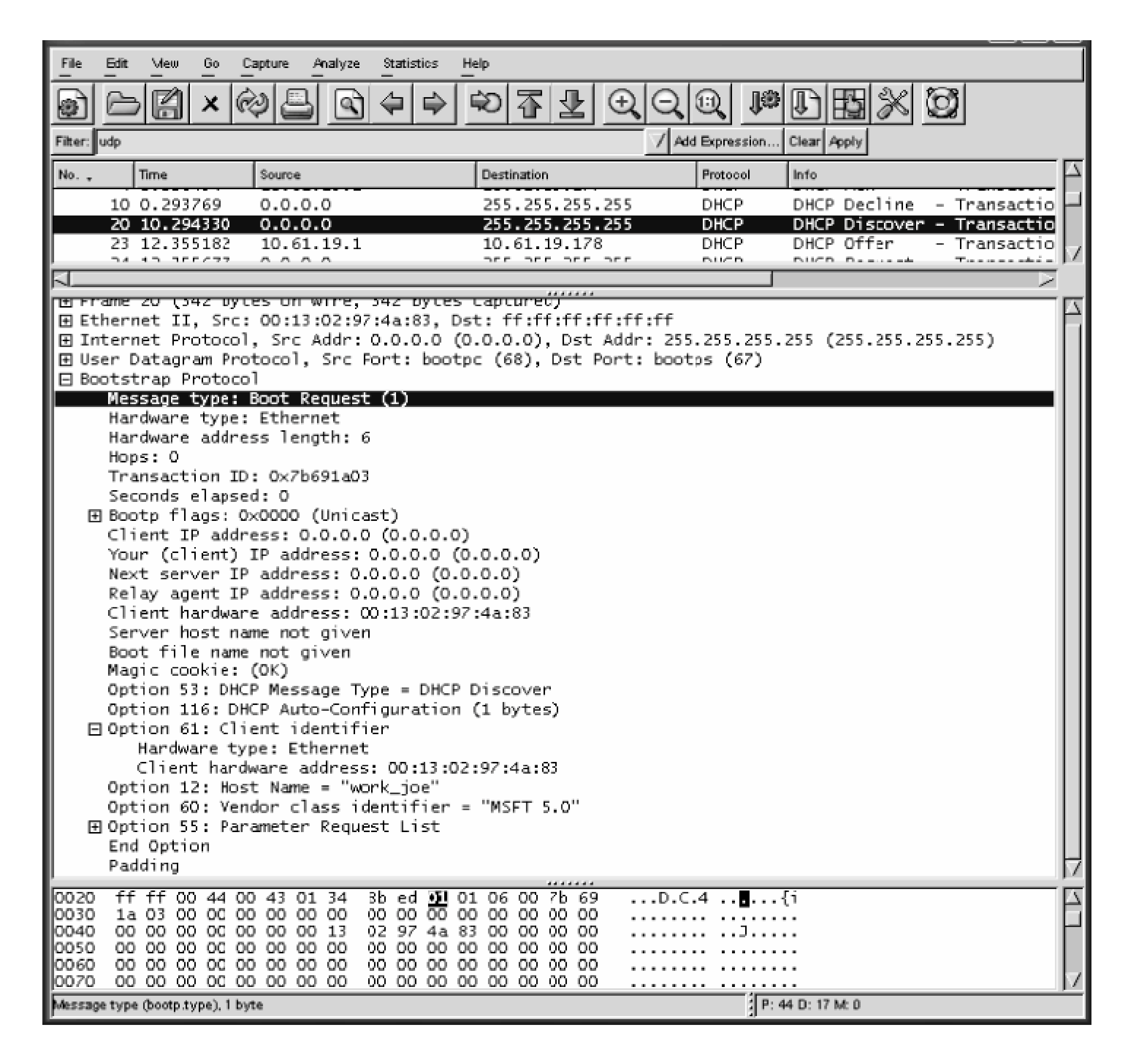



图 10-21

通过前面的截图,可以看见,DHCP 报文用 UDP 承载,客户端使用 UDP 68 端口,服务器端使用 UDP 67 端口。

# 参 考 文 献


［1］ Eric A Hall. Internet 核心协议权威指南. 张金辉译. 北京：中国电力出版社，2002.
［2］ Richard W Stevens. TCP/IP 详解. 卷 1：协议. 范建华译. 北京：机械工业出版社，2000.
［3］ Parker David. TCP/IP 技术大全. 前导工作室译. 北京：机械工业出版社 2000.